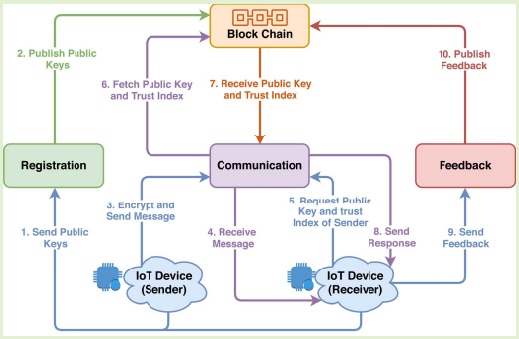
A Scalable Key and Trust Management Solution for IoT Sensors Using SDN and Blockchain Technology

Sufian Hameed, Syed Attique Shah, Qazi Sarmad Saeed, Shahbaz Siddiqui, Ihsan Ali, *Senior Member, IEEE*, Anton Vedeshin, and Dirk Draheim

***Abstract*—Billions of IoT devices and smart objects are already in operation today and even more are expected to be on the network over time. These IoT devices will generate enormous amounts of data that cannot be allowed to transmit on the network without end-to-end encryption or any trust and security mechanism. Currently, we have certificate authorities that certify the identity of a network device by binding its identity with its public key. However, these certificate authorities are centralized in structure and will not be able to individually certify billions of IoT devices entirely. In this paper, we propose that in an SDN-based IoT network, the identities, i.e., public keys and trust indices of IoT devices, can be stored on a blockchain to ensure immutability and tamper-resistance. The paper presents a novel scalable solution for key and trust management of IoT devices in IoT networks, with a successful proof-of-concept that proves the scalability of the proposed solution. The combination of an IoT network along with blockchain technology and software-defined networking (SDN) is effectively demonstrated through simulation that is able to store the public keys of IoT devices on the blockchain and route the network traffic efficiently through SDN. The performance of the proposed solution is evaluated in terms of throughput and access time delay. The results illustrate that access delay and throughput were not affected linearly or exponentially and the proposed solution shows no significant degradation in the performance with the increase in the number of IoT nodes and packets.**



***Index Terms*—Internet of Things, trust management, blockchain, scalability, software-defined networking.**

I. Introduction

USER authentication and identity management have always posed a challenge in traditional internet infrastructure. Conventionally, when two untrusted parties need to communicate securely with each other on the Internet, then a third-party intervention is required to establish the trust between these two communicating parties. This third-party is a certificate authority (CA) whom we have to trust anyway if we need to avail the secured Internet services. Today, Internet users are entirely dependent on CAs because its public key infrastructures (PKIs) are based on certificates and those certificates require the signature of CAs. The CAs in today's hierarchical PKI design are always at risk of being bottlenecks and single points of failure. In fact, they would not be able to scale-up when billions of IoT devices start their operation. Hence, there is a compelling need for highly scalable, distributed third-party service that establish the required level of trust between communicating IoT devices.

The rapid increase in the usage of IoT devices seen in the last decade has stretched the problems related to accessibility, effectiveness of performance, safety, and scalability. Furthermore, an ever increasing number of IoT devices are becoming mobile and, therefore, maintaining their connectivity is a huge challenge. For example, the vision of an Internet of vehicle (IoV) is about providing an infrastructure that supports smart traffic on the roads by maintaining consistent connectivity with all of the many cars moving on streets. IoT devices are

Manuscript received December 9, 2020; revised January 12, 2021; accepted January 12, 2021. Date of publication January 18, 2021; date of current version February 17, 2021. The associate editor coordinating the review of this article and approving it for publication was Prof. Huang Chen Lee. *(Corresponding author: Syed Attique Shah.)*

Sufian Hameed, Qazi Sarmad Saeed, and Shahbaz Siddiqui are with the Department of Computer Science, National University of Computer and Emerging Sciences (NUCES), Karachi 75160, Pakistan (e-mail: k173089@nu.edu.pk; sufian.hameed@nu.edu.pk).

Syed Attique Shah is with the Department of Computer Science, BUITEMS, Quetta 87300, Pakistan (e-mail: attique.shah@buitms.edu.pk).

Ihsan Ali is with the Department of Computer System and Technology, Faculty of Computer Science and IT, University of Malaya, Kuala Lumpur 50603, Malaysia (e-mail: ihsanalichd@siswa.um.edu.my).

Anton Vedeshin is with 3D of Control Systems, Inc., San Francisco, CA 94129 USA (e-mail: anton@3dprinteros.com).

Dirk Draheim is with the Information Systems Group, Tallinn University of Technology, 12618 Tallinn, Estonia (e-mail: dirk.draheim@taltech.ee).

Digital Object Identifier 10.1109/JSEN.2021.3052009

usually connected to the network through an IoT controller which may control several IoT devices in a finite area. When mobile IoT devices move from one location to another, their IoT controllers change continuously because different areas are covered by different IoT controllers. Whenever an IoT device switches its IoT controller, the transition should be smooth without any interruption of service. At the same time, it is important to ensure that the IoT device is still authorized to join the network. For this reason, the registration and access control information of IoT devices have to be stored somewhere far from the edge network.

IoT devices are very limited in computation, power, and resources. We need to utilize them efficiently to conserve their power. As is known, asymmetric encryption algorithms are costly in terms of computation, therefore, IoT devices would consume too much power if the entire communication is performed using asymmetric encryption. Similarly, the initial handshake and sharing of symmetric keys between devices cannot be performed without asymmetric encryption. Secondly, since IoT devices are lightweight and heterogeneous, they might not always be pre-configured to trust the necessary CAs. In order to solve both of these issues, we need a protocol that can offload the handling of trust establishment between IoT devices and IoT services.

Blockchain technology possess features and capabilities that can used to address IoT security problems. The major advantage of incorporating blockchain technology is that it eliminates the need for any centralized third-party authority to validate pieces of information [1], [2]. In a blockchain, trust is established by mutual consensus of multiple third-parties. Moreover, many blockchains are highly available and keep a full ledger on every node. Thus, given the nature of blockchain technology, existing PKIs can be mapped onto blockchain technology which would eliminate our dependency on today's PKI designs and, in particular, their centralized root CAs.

Software-defined networking (SDN) technology allows programmable network configuration that enables resourceful network management in service of enhanced network performance. SDN separates the network components into a control plane and a data plane. The control plane is responsible for routing the packets efficiently, whereas, data plane contains only the flow table entries. SDN controllers-based infrastructures are by design centralized for better control, management and monitoring of network components, which can be its limitation. But since SDN controllers do not have resource constraint issues, they can be scaled with a high-end server as per the computational needs. Furthermore, multiple distributed controllers can also be deployed for better load-balancing. The main benefits of using SDN is that it provides ease of management, centralized control with better security at lower operational costs [3].

SD-IoT (Software-Defined-based Internet of Things) has emerged as a new paradigm in which the well-known scalability issues in IoT are solved by using SDN technology. As the IoT consists of a huge number of devices that generate enormous amounts of data, the current Internet architecture might not be able to support this data requirement. However, SDN is a promising technology that simplifies all the complexities at the controller level and is scalable and efficient [4], [5]. SD-IoT solves the problem of heterogeneity in IoT. Since IoT devices could be a variety of different objects, things or sensors, they may not be data compatible. With SDN-based connectivity, heterogeneous IoT devices can communicate with each other irrespective of the type of device [6]. SDN is also capable of enhancing the security in the IoT network. Much research effort is being proposed to use SDN to provide security in IoT networks in various ways such as to mitigate spoof attacks and DoS attacks [7]. Flauzac *et al.* [8] proposed that SDN controllers can work as border controls that can allow or block IoT devices from connecting across multiple SDN domains. Moreover, SDN enables to move the control login (intelligence) at a central point, so that management, configurations, deployments, etc. operations can be performed easily. If any network decides to move towards SDN deployments whether (LAN, WAN) they can easily update their architecture and deployments. SDN controller can also act as middleware and manage all IoT devices in a network [9]. Intelligent network packet forwarding decisions are taken in the SDN controller which is part of the control plane. The *de facto* standard protocol that is used by devices to communicate with SDN controllers is Openflow. An API for Openflow was first created in 2008 [10]. Once the SDN controller has decided on the forwarding information for network packets, that information is stored on network switches in the form of flow table entries. The flow tables in the switches are part of the data plane. SDN allows network administrators to efficiently program intelligent routing and forwarding algorithms for transmission of network packets. It removes the application-level dependency from the hardware vendors of the routers and switches, allowing network administrators to modify the application logic of routers and switches. The control and configurable nature of SDN makes it applicable to various network topologies [11]. Lately, SDN is being combined with other technologies such as IoT and blockchain. The combination of SDN with IoT and blockchain allows the network to be managed centrally along with configurable functionalities.

A. Motivation

IoT, blockchain, and SDN are emerging technologies that promise to provide a performant and scalable infrastructure for device communications. As we move closer towards the real-time implementations of smart homes and smart cities, it is expected that billions of IoT devices will interconnect to provide numerous services [12]. With an average key size of 2048 bits, a billion keys for the IoT devices would take a 205 – 210 GBs approximately. For scalability purposes, it is always a good idea to store multiple keys in a single block, such as BitCoin store several transactions in any given block. If we decide a block size of around 2 Mbytes, we can store up to 8000 keys in any single block and as an example, the number of blocks required for 1 billion devices would be around 12500 approximately. Thus, considerable attention is required for,

1) Scalable infrastructure which adjusts to the IoT growth and security expectations.

TABLE I
SUMMARY OF IDENTIFIED RESEARCH GAPS THROUGH AVAILABLE LITERATURE

Research Questions	Studies	Summary
Is current IoT network secure?	[21] [22]	Current IoT network solutions suffer from various security issues such as confidentiality, integrity, authentication, non-repudiation, availability and privacy. We need a scalable solution to secure it. Emerging technologies such as SDN and blockchain technology can be useful in providing a secure IoT network.
Can SDN solve the scalability issues in IoT?	[23] [24] [25]	The centralized control and programmable nature of SDN prove to be a solution for managing billions of IoT devices in the future. However, storing the public keys of IoT devices requires a solution that ensure immutability and tamper-resistance, such as blockchain to assist routing the network traffic efficiently through SDN.
Can blockchain solve the scalability issues in IoT?	[26] [27] [28]	The highly distributed *peer-to-peer* nature of blockchain technology allows us to solve scalability issues in IoT networks. Its implementation in real-world applications is still a challenge. Naive blockchain implementations (e.g., based on *proof-of-work* consensus mechanisms without lightweight nodes) must fail.
Can access to blockchain be delegated to fog nodes?	[29] [30]	Fog nodes can be used to access a blockchain and perform authentication tasks for IoT devices. However, not all security threats can be tackled through this approach. For instance, if a fog node is malicious itself then this approach becomes more dangerous for IoT devices.
Can blockchain-based PKI solutions perform better than CA-based PKI?	[31] [32]	In the identified studies, results showed that the certificate verification in IoT devices has been almost three times faster with a blockchain-based PKI as with a CA-based PKI (based on OCSP (Online Certificate Status Protocol) responders).
How to trust IoT devices?	[33] [34]	The shortcomings in available trust proposals for IoT devices is that they do not define the structure of information that would be stored on trust list and nor they identify a mechanism that would calculate trust values of IoT devices.
How to calculate trust indices of IoT devices?	[35] [36]	To accomplish improved key and trust management in the IoT network, feedback and experience values are recommended to calculate trust indices, as they allow a node to evaluate services of another node. Blockchain technology can provide great value to these trust indices if exploited efficiently.

2) Lightweight protocols that can accommodate the limited power, computation and resource requirements of IoT devices.

Through reviewing the available literature, with regards to the research gaps identified and summarized in Table I, it has been recognized that, in order to have a secure and safe IoT infrastructure,

- IoT devices should be able to trust other devices and applications in the network.
- IoT devices should be able to identify malicious devices and applications in the network to minimize security threats.
- A distributed authentication mechanism is required so that IoT devices can verify the identity of other IoT devices and applications on-demand without the risk of a single point of failure.

These identified research gaps drive us to work towards solving the research problems related to implementing a lightweight and scalable key and trust management solution for IoT devices.

The main motivation behind combining SDN with IoT and blockchain is to develop an intelligent, centrally managed and scalable solution that works efficiently for billions of interconnected IoT devices. Due to the centralized control and programmable nature of SDN it has emerged as a promising solution to manage such huge number IoT devices in the future [13]–[15]. In order to support the connectivity of IoT devices through SDN and minimize the amount of data transmission, the SDN-WISE (SDN WIreless SEnsor Networks) [16] solution can be implemented. SDN-WISE minimizes the quantity of information exchanged between IoT sensors and SDN controllers. It provides an exceptional approach towards programmable wireless sensor networks (WSNs). On the other hand, Blockchain offers improved traceability and transparency by providing scalable storage of keys and a framework for the integrity and authentications of keys against any impersonation and forgery [17].

B. Contributions

The main contribution of this paper is a novel scalable solution for the key and trust management of IoT devices in the IoT network with a successful proof-of-concept that proves the scalability of the proposed solution. The public code of our proposed solution is available on Github[1] for readers to evaluate.

For this research in particular, we present,

1) A novel architecture that describes the registration, communication, and feedback phases for IoT devices over a network to accomplish key and trust management.
2) An implementable key and trust management solution of the proposed architecture based on a combination of Contiki [18], [19], Multichain [20], SDNWise [16].
3) The combination of an IoT network along with blockchain and SDN is effectively demonstrated through simulation that is able to store the public keys of IoT devices on the blockchain and route the network traffic efficiently through SDN.

Moreover, the proposed solution is able to store the trust history of IoT devices on the blockchain. As a result, the IoT devices are able to get the information from the blockchain through the SDN controllers and can trust and communicate securely among themselves. The proposed solution is

[1]https://github.com/msufianhameed/scalable-key-trust-solution-for-iot-using-sdn-and-blockchain

evaluated regarding throughput and access time delay. The number of nodes in the IoT network and the number of network packets were systematically increased for evaluation. The results illustrate that access delay times and throughput were not affected linearly or exponentially hence the proposed solution shows no significant degradation in the performance as the number of IoT nodes and packets increases in the IoT network.

C. Paper Organization

The remainder of this paper is organized as follows. Section II presents existing work and finds out how SDN and blockchain technology is currently used to solve scalability challenges in IoT. Moreover, some existing approaches that can allow untrusted IoT devices to trust each other in a network are investigated. Section III thoroughly describes the proposed architecture and design of our scalable solution for key and trust management in an IoT network. Section IV outlines how we developed our prototype and implemented a test-bed on the Cooja simulator. Section V describes different scenarios for our experiments and presents the evaluation results. Finally, conclusions are provided in Section VI.

II. Literature Review

In this section, we review open problems and challenges associated with IoT security and scalability through the help of available literature. We explored how blockchain is used to solve scalability issues in IoT. We have also discussed how current PKIs are not able to handle the keys and certificates of billions of IoT devices due to their intrinsic centralized design. Finally, we identify studies focused on how untrusted IoT devices trust each other by calculating trust indices from their reputation history.

Kouicem *et al.* [21] conducted a top-down survey of security in IoT where they categorized five major IoT applications, i.e., smart grids, healthcare, transportation systems, smart cities, and manufacturing. For each IoT application, they identified the security requirements and challenges. Among the security requirements, the most common were confidentiality, integrity, availability, authentication, and privacy. Common security challenges that were identified were heterogeneity, scalability, mobility and resource limitations. The authors also suggested solutions to these security requirements and challenges using classical approaches as well as new emerging approaches. The classical approaches included using existing PKI solutions and established cryptographic schemes. The new emerging approaches included using SDN and blockchain technologies and they concluded that those can be used to solve security issues in IoT much more efficiently.

Khan and Salah [22] conducted a survey on solving IoT security problems using blockchain. In their paper, they categorized the IoT layers as high-level, intermediate-level and low-level. Within each layer, they identified security risks and provided solutions to those risks. Also, they reviewed various existing solutions to solve IoT security issues using blockchain. Finally, they identified trust management as an open challenge for existing solutions.

Tselios *et al.* [37] provided an overview of security issues in SDN and IoT and proposed that blockchain can significantly solve security issues because it has a purely distributed authentication model, a tamper-proof and immutable data structure and uses distributed verification. Blockchain technology is used to solve various scalability issues in IoT [26]. For instance, by combining IoT and blockchain technology, entire user/device authentication could be handled [29]. Also, the entire key management in a network could be handled when combining IoT with blockchain technology [38]. Instead, it is more efficient to have a centralized server such as Azure IoT edge that has the capability to interact with the blockchain and provide information to IoT devices [39]–[42]. However, one valid argument is that centralized architecture will not be able to handle the scale of the IoT network. For this reason, it is proposed that the SDN technology should be combined with IoT in order to enable a scalable centralized architecture for IoT networks [43].

Various efforts are currently in progress to implement PKI on the basis of blockchain technology. The main motivation behind moving PKI to the blockchain is to obtain decentralization. This would result in protection from single points of failure and transparency in the certificate issuance process. Moreover, certificate revocation and status could be obtained in a decentralized manner. In addition, there is always a risk that CAs can be compromised or that they can misbehave [31], [32], [44]. Durand *et al.* suggested that a *web of trust* model should be used in order to avoid scalability issues [45]. Furthermore, Tewari *et al.* [46] have proposed an entire framework for storing X.509 certificates on the blockchain. In their framework, a device generates a public-private key pair and asks a centralized server to create an X.509 certificate from their public key and publishes it on a blockchain. Other devices can obtain the X.509 certificate from the blockchain and start a trusted and encrypted communication between the each other. The authors have proposed to implement this framework on Multichain [20]. On the other hand, Chen *et al.* [47] are concerned about the audit of PKI certificates. They propose a blockchain-based audit scheme called CertChain to only verify the validity of the certificates such that the certificates that are issued by rogue CAs or are revoked can be detected. Similarly, Madala *et al.* [48] propose not to move the entire PKI to the blockchain, but instead to use it for certificate transparency only. However, moving the entire PKI to blockchain gives us the advantage of not relying on a particular CA.

Mendiboure *et al.* [49] published a paper in which they identified the lack of authentication and authorization in SD-IoV (software-defined Interent of vehicles) as a major shortcoming. They proposed that security should be a primary goal in SD-IoV. For this reason, they proposed a design for a blockchain-based trust management system for SD-IoV using smart contracts. According to their design, all the transactions and exchanges are kept in the blockchain and the nodes are authenticated ensuring integrity, non-repudiation, authentication, availability and confidentiality. Their proposed system aims at enabling a blockchain-based trust establishment exploiting the distributed nature of the SD-IoV control layer.

Furthermore, they proposed a notion of a trust index to manage application trust.

Kataoka *et al.* [33] proposed that a global trust list should be maintained and circulated to IoT devices through a blockchain and SDN. Using this trust list, IoT devices can only trust those devices/services that are whitelisted in that trust list. The shortcomings of this study were that they did not define the structure of information that would be stored on the trust list and they did not identify a mechanism that would calculate trust values of IoT devices.

A feedback mechanism is very important in order to calculate trust values that would establish trust in IoT network [34]. The feedback mechanism collects various parameters from IoT devices when they interact with their neighbors. Various literature resources could be found that create mathematical models that collect feedback parameters from IoT devices and simulate them to obtain a trust value. For instance, Yuan and Li use device-to-device and broker-to-device feedback to calculate global trust values [50]. On the other hand, Namal *et al.* use availability, reliability, irregularities and capacity as parameters to calculate trust values [51]. Similarly, Pietro *et al.* [52] suggested a protocol in which they proposed that the service provider must publish its terms and the service consumer must create obligations when they are about to communicate with each other. At the end of the communication, the service provider can broadcast the obligations as either successful or not. If the service provider broadcasts a successful obligation, then the consumer device is honored and is trusted by other devices on the network [52]. In VANETs (vehicular ad-hoc networks), RSUs (road-side units) that are installed at various locations are responsible for receiving the misbehavior reports from other vehicles and record it on the distributed ledger to blacklist misbehaving devices [53]–[55]. Perhaps, the most generic and common feedback parameters to calculate trust are experience evaluation and reputation evaluation [35], [36].

In the light of blockchain, Truong *et al.* [35] proposed a trust-based IoT model in which a trust platform is responsible for calculating a trust value for IoT devices. The parameters involved in calculating trust values are experience and reputation. Whenever a transaction takes place between two IoT devices, it is followed by a feedback mechanism which collects feedback from the IoT devices on the transaction that has occurred between them. The feedback can be positive or negative based on the value that the transaction provided to the IoT device and is used to simulate the mathematical models that calculate the experience and reputation values. Finally, the values of experience and reputation are combined to calculate the value of trust. This trust value can be used to establish a smart contract between the two IoT devices. In this way, the IoT device can only be trusted if the trust value exceeds a certain threshold defined in the smart contract.

The existing literature on the usage of SDN and Blockchain for IoT security and trust management lacks implementable solutions that can offer scalability attributes and lightweight protocols, keeping in view the IoT devices growth and resource constraints respectively. Comparatively, in our proposed solution we are maintaining trust values on the blockchain that can provide a complete transparent trust history of any node through the SDN controllers, keeping in view the scalability and accommodation of the limited resources of IoT devices. Moreover, our proposed solution shows no significant degradation in the performance even with the increase of IoT nodes and packets.

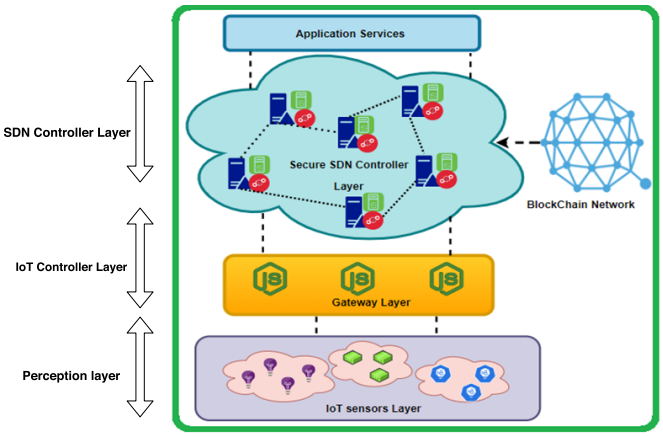


Fig. 1. Framework for the proposed scalable solution.

III. System Architecture and Design

The proposed scalable solution for IoT devices is working on the framework as shown in Figure 1. The framework is based on a four-layers i.e., perception layer (IoT sensors), IoT controller, SDN layer, and the application layer. The aim of SDN is to make networks agile, flexible, and smart. SDN decouples the network control from the data forwarding with the help programming interface that overcomes the limitation of the IoT controller of the vendor-dependent solution. Blockchain network is added in this framework for vertical and horizontal scalability, security, and the fundamental limitation of SDN due to its central control or single-point failure. Blockchain is a decentralized paradigm where single-point failure is impossible. Our solution reflects both features of blockchain that are security and scalability.

In our proposed solution, SDN is used to route the network traffic and blockchain technology is deployed to store the keys and trust history of the IoT devices in the network. The architecture design for the proposed solution is broken down into the three parts, i.e., registration, communication, and feedback. In *registration*, the IoT device generates the key-pair and publishes the public key on the blockchain via an SDN controller. In *communication*, the IoT devices are able to trust and securely communicate with each other. In *feedback*, IoT devices post their communication experience on the blockchain through the SDN controller.

For the sake of simplicity of our design, we assume that the communication between the SDN controller and the blockchain is secure since the SDN controller and the blockchain node are running on the same machine. Also, we assume that all IoT devices trust the IoT controller through some web of trust. This means that the identity and public key of the IoT controller are signed by a root certificate authority and all IoT devices trust that root certificate authority. Therefore, IoT devices can verify the authenticity of an IoT

controller by sending a challenge to the IoT controller. If the IoT controller is able to verify the challenge, then IoT devices will be sure that the IoT controller can be trusted. We also assume that the communication between the IoT controller and the SDN controller is secure through some pre-shared symmetric keys. It is safe to assume pre-shared symmetric keys between the SDN controller and the IoT controller because both these entities are static and will not move as the IoT devices do. For example, in IoV, RSUs (Road-Side Units) are installed at various locations on the road that work together as the IoT controller. We are also aware of the fact that it is not feasible for IoT devices to use heavyweight key generation algorithms to generate keys. For instance, generating a 2048-bit RSA key-pair might consume a fair amount of power in power constrained IoT devices. We therefore, require a lightweight key generation algorithm. However, we are still determined to use either 2048-bit RSA or AES-128 for asymmetric encryption due to their robustness. One possible solution is to delegate the key generation task to IoT controllers that are working as fog in our IoT network. After generating a key-pair for an IoT device, the IoT controller will need to securely transfer the private key to IoT devices. This can be done according to the following sequential steps:

1) IoT device contacts the IoT controller to join IoT network
2) IoT controller sends back a certificate to IoT device
3) IoT device will verify the signature on the certificate using the public key of the certificate authority that it trusts
4) IoT device will create a session key, encrypt it with IoT controller's public key and send it to IoT controller
5) IoT controller will generate the key-pair, encrypt the key-pair with the session key and send it back to IoT device
6) IoT device will decrypt the key-pair using the session key

Figure 2 represents the message structure of the proposed framework. The message structure's total size is 256 Byte because it encrypts with an RSA 2048-bit public key. The message structure contains the message header, the token, signature, and the flag. The message header contains the message of IoT nodes. The token header contains the MAC of a particular message. The signature header includes the signature of a particular IoT node, and the flag represents the end of the message. Since, the size of the message is too large to transmit it into the IoT network, so the message is fragmented into four chunks of 64 bytes in the simulated environment.

A high-level design of the entire architecture is described in graphical abstract. It shows an end-to-end flow on how IoT devices will interact in the registration, communication and feedback processes in order to communicate securely with other IoT devices in the network. The details of all the three processes are discussed in detail in the subsequent sections.

A. Registration

The architecture and design for the registration process of IoT devices is shown in Figure 3. Each step is further elaborated in the following sections.

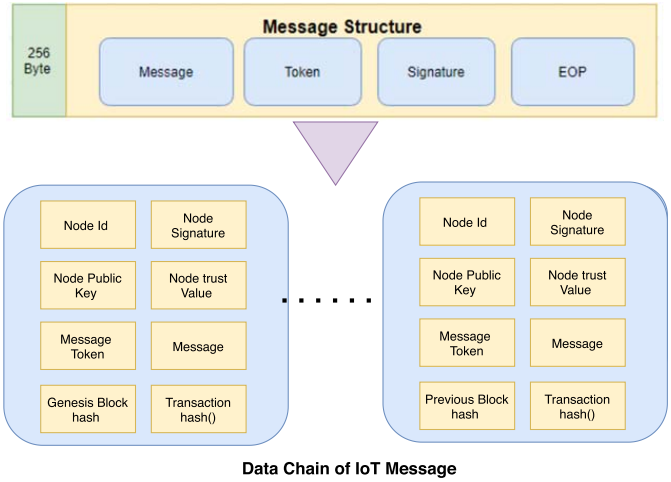


Fig. 2. Message structure of the proposed framework.

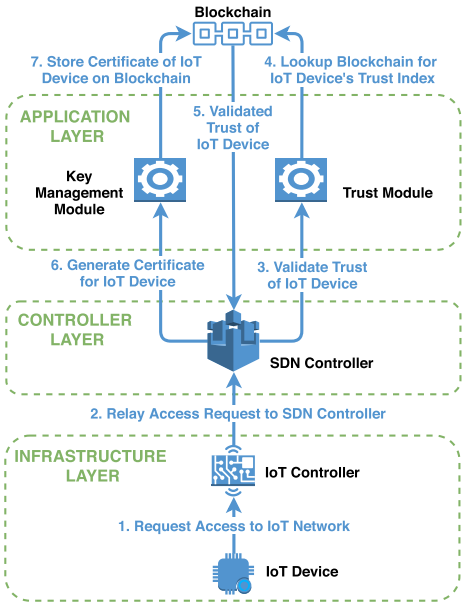


Fig. 3. Registration process of IoT devices requesting access.

1) Request Access to IoT Network: Any IoT device that requires access to the IoT network, will have to request the SDN controller to publish its public key on the blockchain. Since IoT devices do not have the ability to communicate directly with SDN controllers, they will communicate with an IoT controller to get access to the IoT network. The IoT device will generate a public-private key pair and then digitally sign a message using its private key to create a message digest. It will then send the encrypted message digest, message and public key to the IoT controller. Later on, any other service can verify the authenticity of the IoT device by taking a hash of the message and compare it with the message digest. Also, end-to-end encryption can be performed as services can encrypt the response by using the public key of the IoT device.

2) Relay Access Request to SDN Controller: Since the IoT controller does not have the ability to publish data on blockchain, therefore, it will pass the provided information to an SDN controller. SDN controller will first verify the

authenticity of the IoT device by taking a hash of the message and compare it with the message digest. In the next step, the SDN controller will decide whether the IoT should be trusted or not.

3) Validate Trust of IoT Device: The IoT devices that request access to the IoT network may or may not be malicious. For this reason, the SDN controller will need to validate if the IoT device can be trusted or not. The trust module installed on the SDN controller will be responsible for this job. Trust module is an application service which is programmed to handle the incoming IoT device access requests. Also, it has access to the blockchain. In general, the trust module will have the following two functions:

(a) Lookup blockchain for IoT device's trust index.
(b) Add IoT device's trust index on blockchain.

For simplicity, we will initially focus on the first function and assume that the trust indices of IoT devices are already present on the blockchain. In the next step, the trust module will search the blockchain to obtain a trust index of the IoT device in order to decide if the IoT device can be trusted or not.

4) Lookup Blockchain for IoT Device's Trust Index: Trust index is a value that is ranging between 0 and 1. If the trust index is close to 0, then the IoT device should not be trusted, whereas, if the trust index is close to 1, the IoT can be trusted. While programming the trust module, we can set up the threshold value so that only those IoT devices can be trusted whose trust index is either unknown or greater than the threshold value. In order to persistently store the trust indices of IoT devices, we require a scalable database that can be accessed by all SDN controllers. For this reason, we are using a permissioned blockchain as a distributed database. All the SDN controllers will be subscribed to this blockchain. Moreover, they will be able to publish blocks of data on this blockchain. The trust module will traverse the blockchain in reverse order (i.e. from the latest block till genesis block) and search for the block that contains the latest trust index which is owned by the public key of the IoT device which is requesting the access. If the data block is found, then the trust module will read the trust index from that block and then decide if the trust index passes the threshold value or not. If it does not pass, then access is denied, otherwise, the next step is followed.

5) Validated Trust of IoT Device: If the trust module was successfully able to validate the trust index of the IoT device, then it will ask the SDN controller to proceed with the processing of the incoming request. In the next step, the key management module will generate a symmetric key for the IoT device which will be used by IoT devices to connect to the IoT network and communicate securely.

6) Generate Certificate for IoT Device: Key management module is another application service that is running on the SDN controller. The purpose of this service is to generate a certificate for the IoT device. The certificate contains the public key and the address of the IoT device. Since the identity of the IoT device was already validated in the second step, therefore, the SDN controller does not need to do it again. Instead, the SDN controller just needs to sign the generated certificate by taking the hash of the certificate and sign that hash with its private key. Then, this digital signature needs to be appended to the certificate along with the public key of the SDN controller.

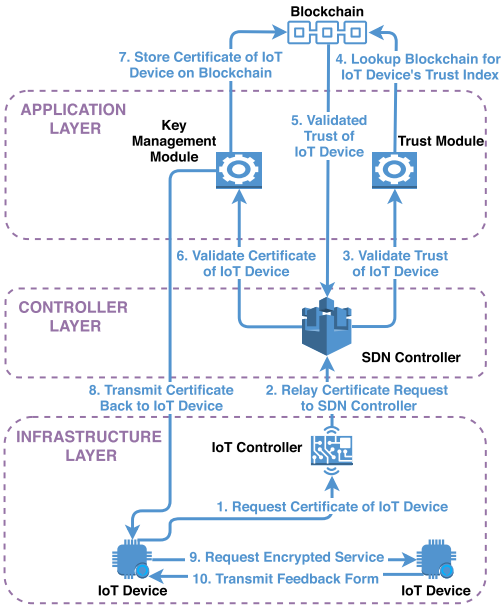


Fig. 4. Communication process of IoT devices.

7) Store Certificate of IoT Device on Blockchain: The final step for the SDN controller is to publish the signed certificate of the IoT device on the blockchain. As a result, any IoT device that trusts the public key of the SDN controller can trust all the other IoT devices whose certificates are signed by that SDN controller.

B. Communication

The design and architecture for the communication process of IoT devices is shown in Figure 4. Each step is further elaborated in the subsequent sections.

1) Request Certificate of IoT Device: Any IoT device (i.e. client IoT device) that wishes to receive a service from another IoT device (i.e. server IoT device) will first need to obtain the certificate of server IoT device. The certificate contains the public key of the server IoT device and a digital signature from an SDN controller that certifies that the public address and public key of the server IoT device are bound correctly. The client IoT device will know the public address of the server IoT device. Client IoT device will send its certificate query to the IoT controller by providing it the public address of the server IoT device as a key.

2) Relay Certificate Request to SDN Controller: Since the IoT controller does not have the ability to access the blockchain, therefore, it will relay the incoming request to the SDN controller.

3) Validate Trust of IoT Device: Now, the trust module will lookup the trust index of the server IoT device and determine if the server IoT device can be trusted or not. It will send this information back to the client IoT device so that it can suspend establishing connection with the untrusted server IoT device. This step is similar to the step that was described earlier in the registration process. The only difference is that instead of

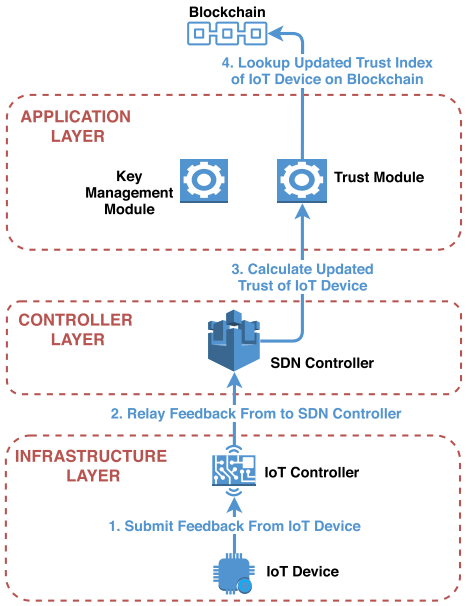


Fig. 5. Feedback process of IoT devices.

looking up the public address of the client IoT device, we will now lookup the public address of the server IoT device.

4) Lookup Blockchain for IoT Device's Trust Index: In this step, the latest trust index of the server IoT device will be fetched from the blockchain by the trust module. This step is similar to the step that was described earlier in the registration process.

5) Validated Trust of IoT Device: If the trust index of the server IoT device exceeds a certain threshold, then the trust module will validate it and allow the SDN controller to proceed to the next step and fetch the certificate of server IoT device.

6) Fetch Certificate of IoT Device: Now, the SDN controller will ask the key management module to fetch the certificate of the server IoT device. Since the key management module has access to the blockchain, therefore, it will formulate a request to the blockchain and fetch the certificate.

7) Lookup Certificate of IoT Device on Blockchain: The request sent by the key management module in the previous step will be executed in this current step. The public address of the server IoT device will be looked up on the blockchain.

If the certificate of server IoT was successfully stored on the blockchain at the time of registration, then that certificate would be found in this step.

8) Transmit Certificate Back to IoT Device: In this step, the certificate of the server IoT device that is found needs to be transmitted back to the client IoT device. The certificate will contain the public key of the server IoT device and will be bound by its public address. Along with this, the certificate would be digitally signed by the SDN controller that issued it. The validation of the certificate can easily be performed at the client IoT device by taking the hash of the certificate, then decrypt the digital signature by the public key of the SDN controller and then finally match them together.

9) Request Encrypted Service: Finally, the client IoT device can trust the server IoT device and communicate securely by encrypting the requests with the public key of the server IoT device. Later on, they may share a symmetric key among themselves to engage in a long-term session.

10) Transmit Feedback Form: The server IoT device needs to send a feedback form to the client IoT device so that the client IoT device can, later on, report his experience with the server IoT device. We will discuss how this feedback will affect the trust index of the server IoT device later in the feedback process. For now, we simply need to send a feedback form to the client IoT device. For feedback form, we have generated a token at the server IoT device at the end of the communication. The server IoT device with digitally sign the token by taking a hash of the token and encrypt it with its private key and then send that token back to client IoT device along with the digital signature of the token. Upon submitting the feedback, the client IoT device will redeem that token to allow feedback to be submitted.

C. Feedback

The design and architecture for the feedback process of IoT devices is shown in Figure 5. Each step is further elaborated in the subsequent sections.

1) Submit Feedback Form of IoT Device: Once the client IoT device has finished its engagement with the server IoT device, it will then report its experience back to the SDN controller so that the trust index of the server IoT device can be calculated. The experience can be a value ranging from 0 to 1. If the value of experience is close to 0, then it means that the client IoT device had a bad experience with the server IoT device. Whereas, if the value of experience is close to 1, then it means that the client IoT device had a good experience with the server IoT network. The value of experience is based on various factors. Usually, it is a better approch if the client IoT device successfully made proper use of the information that was provided by the server IoT device. The historic collection of experience values are used to form a reputation value. We will discuss later how experience and reputation values are used as parameters to calculate the trust index. Recall that, the server IoT device transmitted the feedback form in the shape of a token back to the client IoT device. Now, when the client IoT device submits the experience back to the SDN controller, then the feedback token is redeemed by the client IoT device. Once the feedback token is redeemed and feedback is saved, then it cannot be changed.

2) Relay Feedback Form to SDN Controller: Since the IoT controller does not have the ability to access the blockchain, therefore, it will relay the information to the SDN controller. SDN controller will then pass the request to the trust module that will calculate the updated trust index of the server IoT device in the next step.

3) Calculate Updated Trust Index of IoT Device: Trust module is also responsible for accepting the feedback request from the client IoT devices and use the experience value to calculate the new trust index of the server IoT device. First, it will fetch the existing trust index of the server IoT device from the blockchain. The data fetched from the blockchain will contain the trust index along with some historic reputation of the server IoT device. We appended the experience value in the historic

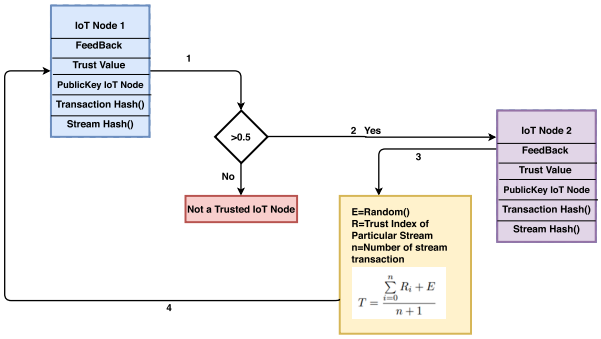


Fig. 6. Trust value calculation scenario.

reputation and let our model calculate the new trust index. The calculation model of trust index is described below in Equation 1,

$$T = \frac{\sum_{i=0}^{n} R_i + E}{n+1} \tag{1}$$

T is the trust index of the IoT device that is being calculated. n is the number of historic reputation values that are retrieved from the existing trust index record. R_i is the historic reputation value at a particular index. E is the experience value of the IoT device that is submitted by the client. In the simulation this experience value is calculated with the help of random function ranging from 0-to-1, but in real scenario this experience value depends on various real time parameters. This is a simplified version of the formula that was proposed by Truong *et al.* [35]. The only difference is that they were using ΔR in their formula to calculate the trust value at any given instance. In our case, we do not have a calculated value of ΔR stored on blockchain. Infact, it is easier for us to fetch the previous values of R_i from the blockchain and use them in our calculation. Also, we will not require the complexity of decaying ΔR since we are not storing it anywhere. Instead, we have kept a timestamp for validity with all the historic values of R_i and make sure that trust value is calculated by using R_i that are within a particular timestamp.

To elaborate the trust value calculation we present a scenario in Figure 6. Suppose "IoT Node 1" sends an encrypted message to "IoT Node 2". First, the trust module checks whether the IoT Node is trusted or not. If it is trusted, then "IoT Node 2" decrypts the message with the public key and submits the feedback to the "IoT Node 1". The chain record is calculated according to Equation 1 and submitted to the "IoT Node 1" chain for further executions.

4) Store Updated Trust Index of IoT Device on Blockchain: The final step is to store the updated trust index of the server IoT device on the blockchain. Trust module will be responsible to publish the trust index of the server IoT device once it has calculated the updated trust index.

IV. Testbed and Implementation

Our implementation is simulated on the Cooja [19] simulator which is an open-source Contiki [18] network simulator. Cooja simulator is very popular in WSN (Wireless Sensor Networks) research community. It is used in the simulation of wireless sensor networks and benchmarks the performances when multiple devices are communicating with each other in a wireless sensor network [56], [57]. Moreover, Cooja simulator allows us to simulate the communication of Contiki OS-based IoT devices in a wireless sensor network using lightweight network-level protocols such as CoAP and 6LowPAN. It allows the network of Contiki motes to be simulated and can be emulated at hardware levels. Contiki motes are basically IoT devices the run on Contiki OS. Cooja simulator is implemented in Java and runs on Linux machines. It is used to simulate a network of IoT devices and inspect the exchange of data packets on the underlying network layer.

On top of Cooja simulator, SDN-WISE (SDN WIreless SEnsor Networks) [58] is also simulated. We can deploy a virtual wireless sensor network using Cooja and SDN-WISE. SDN-WISE is an SDN solution for wireless and infrastructureless networks. It allows wireless devices based on the IEEE 802.15.4 physical and MAC layers to communicate with each other on SDN-based networks. It contains sinks and different nodes in its network. Sinks are connected to the network infrastructure, whereas, nodes are wireless. All data packets from the IoT nodes are transmitted to the controller via sinks. It also contains a forwarding layer that handles the incoming packets as specified inflow tables. The control plane is responsible to update the flow tables via the forwarding layer.

The simulation of SDN-WISE is dependant on the Cooja simulator. The simulated SDN-WISE is also an open-source Java-based application that is compatible to run on top of the Cooja simulator. Cooja and SDN-WISE, when combined together, provide us a virtual SD-IoT network on which we can perform our experiments and evaluate them.

Finally, MultiChain [20] is used as the blockchain. MultiChain is a fork of Bitcoin and can be used to deploy a permissioned blockchain. MultiChain is independent of a platform, but since we deployed Cooja simulator and SDN-WISE on Linux and our SDN controller resides on Linux, therefore, we have installed MultiChain on Linux as well.

According to our architecture, the key management module and trust module will require access to the blockchain. Since the key management module and trust module run on top of the SDN controller, therefore, we have connected the SDN controller with blockchain. Both key management module and trust module can access the blockchain via SDN controller. Note that, Cooja simulator and SDN-WISE are Java-based applications and we use the MultiChain PHP API to invoke remote procedure calls to interact with MultiChain.

For the test bed, we have a Linux machine on which the Cooja simulator, SDN-WISE, MultiChain and MultiChain PHP API are installed. These four components are sufficient for us to perform our experiments. Figure 7 shows a block diagram of the four components and their interaction with each other. The sensor nodes were deployed on IEEE 802.15.4 boards with 8 kB RAM + 256 kB Flash memory. The hardware and software configuration of the control system that we used for performing our experiments are shown in Table II.

The sequential work flow for the test bed is as following:

1) When IoT network starts, generate an RSA-512 bit keypair and an RSA-2048 bit keypair. A 512-bit key is used

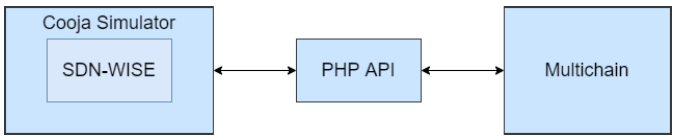


Fig. 7. Block diagram of testbed components.

TABLE II
CONFIGURATION OF THE CONTROL SYSTEM

Item	Specification
Processor	Core(TM) i5-5300U 2.30 GHz
Cores	2 cores allocated out of 4
RAM	3 GB
Storage	10 GB of mSATA SSD
OS	Ubuntu 18.04.4 LTS
MultiChain	2.0-alpha-5
Java	Version 8 OpenJDK
PHP	7.2

to sign tokens of feedback forms, whereas 2048 bit key pair is used to encrypt/decrypt the entire message.

2) Publish both the public keys on blockchain.
3) Select a node that sends message to another node.
4) Generate a token, take a hash of it and encrypt it with the 512 bit private key of the sender node.
5) Generate the message.
6) Combine the message and token in a single payload and fragment it in chunks of 64 bytes.
7) Encrypt each chunk with the 2048 bit private key of the sender node and send it.
8) When the message is received at the destination, fetch the public key of the sender node from blockchain, decrypt each chunk with it and re-assemble the fragmented message.
9) Then fetch the calculated trust index from the reputation history stored on blockchain. Reputation history records must be within a certain time period and its decrypted token signature must match with the hash of the token that it generated earlier.
10) If the trust index does not surpass a certain threshold, then ignore the message.
11) Publish the experience with the sender IoT device on blockchain.

The pseudocode for the key submission process is presented in Algorithm 1. This is an important segment of the code because it identifies the area where the public key of IoT device is submitted to blockchain. We have also logged the time taken by this operation so that we can measure the access time delays and throughput in our evaluations. The pseudocode shown in Algorithm 2 identifies the phase where the message is encrypted and sent to the IoT device. It represents the overall process of IoT devices encrypting the message using their private keys, breaking the messages into smaller fragments and transmitting them over the network. Similarly, pseudocode shown in Algorithm 3 identifies the area where the message is received and decrypted. This is where IoT devices are fetching the public keys of other IoT devices and then decrypting the received message. Algorithm 4 presents the pseudocode for the code segment where feedback is submitted to blockchain. It represents the process where IoT devices are fetching the trust index from the blockchain, deciding whether to trust the IoT device from which the message has been received and posting the communication experience back to the blockchain.

Algorithm 1 Key Submission

Input: *Node ID, PubKey 512, PubKey 2048*
Output: *(Success/Failure)*
Steps:
1: **for** Every node in IoT network **do**
2: $nodeID$ = Get node_ID of node
3: $pubKey512$ = Get node's encoded 512-bit RSA public key
4: $pubKey2048$ = Get node's encoded 2048-bit RSA public key
5: PublishOnBlockchain($node\_ID$, $pubKey512$)
6: PublishOnBlockchain($node\_ID$, $pubKey2048$)
7: **end for**

Algorithm 2 Encrypt and Send Message

Input: *Plain Text Message, sender Private key*
Output: *Yes/No (successfully encrypted and sent)*
Steps:
1: $message$ = Set message to send
2: $token$ = Generate random UUID
3: $tokenHash$ = GenerateHash($token$)
4: $privKey512$ = Get 512-bit RSA private key of message sender node
5: $encryptedTokenHash$ = Encrypt($tokenHash$, $privKey512$)
6: $text = message + token + encryptedTokenHash$
7: $privKey2048$ = Get 2048-bit RSA private key of message sender node
8: $encryptedText$ = Encrypt($text$, $privKey2048$)
9: $payloadBytes$ = ConvertToBytes($encryptedText$)
10: **for** Payload bytes remaining to send **do**
11: Send 64 bytes of payload to network
12: **end for**

V. EVALUATION

The proposed solution is evaluated regarding throughput and access time delay. The throughput will state the number of bytes of data that were successfully transmitted over the network per second when the trust index of IoT devices was fetched from the blockchain. Whereas, the access delay time will determine the time taken by the operation. The evaluation method captures the throughput and access time delay of the following six operations that are aligned with the proposed solution:

1) Key submission on blockchain
2) Key retrieval from blockchain
3) Encrypt and send messages
4) Trust index retrieval from blockchain
5) Decrypt message
6) Feedback submission on blockchain

Algorithm 3 Decrypt Message

Input: *receiver ID, Encrypted Message*
Output: *Plain Text Message*
Steps:
1: $encodedKey$ = Get encoded public key of message sender node from blockchain
2: $decodedKey$ = Decode($encodedKey$)
3: $decryptedMessage$ = Decrypt($message$, $decodedKey$)

Algorithm 4 Feedback Submission

Input: *senderID*
Output: *Feedback of trust value is submitted*
Steps:
1: $trustIndex$ = Get trust index of message sender node from blockchain
2: **if** $trustIndex > threshold$ **then**
3: Accept packet and use data
4: $receivedToken$ = Get token from received packet
5: $receivedTokenSignature$ = Get token's signature from received packet
6: $experience$ = Receiver node's experience with sender node's data (either 0 or 1)
7: PublishOnBlockchain($receiverNode\_ID$, $experience$, $receivedToken$, $receivedTokenSignature$)
8: **else**
9: Ignore packet
10: **end if**

The number of IoT devices and packets along with the captured throughput and access time delay are changed three times. Finally, the average of the captured values is taken and used later for our conclusion. We are assuming that the IoT devices in our scenario are not mobile. Moreover, we are assuming that all the IoT devices are registered on the network when the network starts and no IoT device leaves the network until the network is stopped. Also, we are not monitoring the CPU, memory, and battery of IoT devices in our evaluation.

A. Key Submission on Blockchain

The first operation in our solution is the key submission on blockchain. When the network starts and an IoT device registers itself on the network, it generates two public-private keypairs using RSA-512 and RSA-2048. The reason for having two public-private keypairs is that RSA-512 keypair is used to sign and verify the signature of token that an IoT device provides with the feedback form. Whereas, RSA-2048 keypair is to encrypt and decrypt the entire payload that is sent during IoT device communication. We are interested in capturing the throughput and access time delay when IoT devices publish their public keys on the blockchain through the SDN controller. We have changed the number of IoT nodes and packets and captured the throughput and access time delays. Each node submits exactly two public keys on the blockchain. Since each node submits exactly two public keys on the blockchain, changing the number of packets will require us to change the working code, so we have not performed it. The results of this operation are given in Table III.

TABLE III
KEY SUBMISSION ON BLOCKCHAIN WHEN CHANGING IOT NODES

No. of Nodes	Throughput (bytes/s)	Access Time Delay (ms)
5	3630	501.055
10	4843.555	355.787
15	4143.311	265.645
20	2997.288	253.761
25	2424.277	282.269

TABLE IV
KEY SUBMISSION ON BLOCKCHAIN WITH RESPECT TO REQUESTS

Number of Keys	Throughput (requests/s)	Processing Delay (ms)
100	1.53	648.22
500	6.47	155.574
1000	12.28	81.459
2000	21.0	47.4475
3000	18.33	27.32
3050	14.98	25.02

For testing the scalability of the blockchain on our proposed solution, we initially increased the key submission requests per node with delays of 600ms, 120ms, 60ms, 30ms and 10ms, to capture and evaluate the throughput and access time delays. The results of this operation are given in Table IV. The result clearly indicates that the throughput of the request increases and the processing delay time between the keys decreases when the number of requests is increased.

B. Key Retrieval From Blockchain

The second experiment in our solution is key retrieval from blockchain. When an IoT device needs to send a message to another IoT device, then it needs to fetch the public key (i.e. 2048-bit public key) of the other IoT device, so that it can encrypt the message with that public key. Once again, we change the number of IoT nodes and capture the throughput and access time delays. The results of this operation are defined in Table V. Furthermore, we change the number of key retrieval requests with increased delay that is 600ms, 120ms, 60ms, 30ms to capture the throughput and access time delays. The results of this operation are defined in Table VI. The result clearly indicates that the throughput of the request is increased and the processing delay time decreases at inconstant rate when the number of requests is increased at a certain amount. To evaluate the solution with increasing number of packets, we use 10 nodes in the network and increase the number of packets to be delivered. The same nodes fetch the key from the blockchain to send the message to 10 other nodes. The results of this experiment are listed in Table VII.

C. Encrypt and Send Messages

The third operation in our solution is to encrypt and send messages. After successfully fetching the public key from the blockchain, the IoT device is ready to create a payload, encrypt it and send it over the network so that the message can reach its destination. The throughput and access time delay results when the number of nodes and packets are changed are given in Table VIII and Table IX respectively.

TABLE V
KEY RETRIEVAL FROM BLOCKCHAIN WHEN CHANGING IoT NODES

No. of Nodes	Throughput (bytes/s)	Access Time Delay (ms)
5	1004.93	92.381
10	1154.074	62.78
15	1136.351	79.62
20	1215.698	79.22
25	1210.032	79.906

TABLE VI
KEY RETRIEVAL ON BLOCKCHAIN WITH RESPECT TO REQUESTS

Number of Keys	Throughput (requests/s)	Processing Delay (ms)
100	0.67	1449.82
500	2.18	475.662
1000	3.03	495.697
2000	6.52	143.8305
3000	5.00	172.147
3050	3.00	292.804

TABLE VII
KEY RETRIEVAL FROM BLOCKCHAIN WHEN CHANGING PACKETS

No. of Packets	Throughput (bytes/s)	Access Time Delay (ms)
50	1154.074	62.78
100	1165.238	62.843
150	1166.625	62.251
200	1168.97	62.68
250	1171.414	62.984

TABLE VIII
ENCRYPT AND SEND MESSAGES WHEN CHANGING IoT NODES

No. of Nodes	Throughput (bytes/s)	Access Time Delay (ms)
5	106.731	445.58
10	107.938	430.686
15	107.916	433.186
20	107.916	432.326
25	107.916	431.073

TABLE IX
ENCRYPT AND SEND MESSAGES WHEN CHANGING PACKETS

No. of Packets	Throughput (bytes/s)	Access Time Delay (ms)
50	107.938	430.686
100	107.288	424.896
150	107.238	422.544
200	107.115	420.651
250	107.084	420.064

D. Trust Index Retrieval From Blockchain

The fourth operation in our solution is trust index retrieval from blockchain. When an IoT device receives a message from another IoT device, then the first thing it needs to do is to fetch the trust index of the IoT device from which the message is received. Based on a certain threshold value of the trust index, the IoT device will decide whether to drop the packet from the untrusted IoT device or proceed further with its decryption. We have changed the number of nodes and captured the throughput and access time delay. The results of this operation are provided in Table X. Similarly, we have changed the number of trust retrieval requests with increased delay that is 600ms, 120ms, 60ms, 30ms to capture the throughput and access time delays. The results of this operation are shown in Table XI. For this experiment, we have used 10 nodes in the network and increased the number of packets in each test run. The results of this operation are provided in Table XII.

TABLE X
TRUST INDEX RETRIEVAL FROM BLOCKCHAIN WHEN CHANGING IoT NODES

No. of Nodes	Throughput (bytes/s)	Access Time Delay (ms)
5	2.574	656.246
10	2.950	467.073
15	2.902	561.373
20	3.101	582.053
25	3.082	541.313

TABLE XI
TRUST INDEX RETRIEVAL ON BLOCKCHAIN WITH RESPECT TO REQUESTS

No. of Requests	Throughput (requests/s)	Processing Delay (ms)
100	0.47	2187.99
500	1.32	773.444
1000	1.93	513.441
2000	3.43	277.303
3000	3.33	278.974
3050	2.32	433.843

TABLE XII
TRUST INDEX RETRIEVAL FROM BLOCKCHAIN WHEN CHANGING PACKETS

No. of Packets	Throughput (bytes/s)	Access Time Delay (ms)
50	2.95	467.073
100	2.974	473.546
150	2.978	473.524
200	2.985	475.411
250	2.989	475.458

E. Decrypt Message

The fifth operation in our solution is to decrypt the message. When an IoT device receives a message from another IoT device and its trust is passed, the IoT device will decrypt the message using its private key so that it can obtain the information from the other IoT device. In this case, there will be no throughput because this is not a send operation and is only a receive operation. We have changed the number of IoT nodes to capture the access time delay only. The results of this operation are given in Table XIII.

For the next experiment, we have focused to find out the access time delay in the message decryption process using the same strategy of changing the number of packets in each test run. The results of this operation are provided in Table XIV.

F. Feedback Submission on Blockchain

The sixth and final operation in our solution is feedback submission on blockchain. When an IoT device successfully

TABLE XIII
DECRYPT MESSAGE WHEN CHANGING IOT NODES

No. of Nodes	Access Time Delay (ms)
5	3.298
10	1.266
15	3.933
20	2.46
25	4.16

TABLE XIV
DECRYPT MESSAGE WHEN CHANGING PACKETS

No. of Packets	Access Time Delay (ms)
50	1.266
100	1.166
150	1.264
200	1.233
250	1.238

TABLE XV
FEEDBACK SUBMISSION ON BLOCKCHAIN WHEN CHANGING IOT NODES

No. of Nodes	Throughput (bytes/s)	Access Time Delay (ms)
5	103.588	72.542
10	129.730	45.66
15	122.923	60.333
20	132.641	60.433
25	131.144	54.68

receives and decrypts a message from another IoT device, then it can submit feedback on the blockchain that represents its experience with that IoT device from which the information is received. This operation is important because a sum of these feedback values is used by the current and other IoT devices in the future to calculate the trust index. We have changed the number of nodes and requests again to obtain the throughput and access time delay. The results of this operation are defined in Table XV. The results of the number of requests operation for feedback are defined in Table XVI. The result clearly indicates that the throughput of the request is decreased and the access delay time increases when the number of requests is increased at a certain amount that means the system is scalable at a certain level. Similarly, our final experiment is to find out the throughput and access delay time when IoT devices submit their feedback on the blockchain. This feedback includes information about the positive or negative experience that the IoT device had while communicating with another IoT device. We find out how many bytes of data were transmitted over the network per second and how much time was taken in this operation when the IoT devices submitted their feedback on the blockchain. For this experiment, we have taken 10 nodes in the network and increased the number of packets gradually in each test run. The results of this operation are defined in Table XVII.

G. Discussion

The conducted experiments show that even if the number of IoT nodes or the number of packets are increased,

TABLE XVI
FEEDBACK SUBMISSION ON BLOCKCHAIN WITH RESPECT TO REQUESTS

No. of Feedback	Throughput (feedback/s)	Processing Delay (ms)
100	0.35	2,873.60
500	1.08	924.6
1000	1.72	588.45
2000	2.13	461.47
3000	1.67	441.45
3050	1.65	554.75

TABLE XVII
FEEDBACK SUBMISSION ON BLOCKCHAIN WHEN CHANGING PACKETS

No. of Packets	Throughput (bytes/s)	Access Time Delay (ms)
50	129.730	45.66
100	132.178	45.26
150	132.226	45.184
200	132.741	46.046
250	133.112	46.252

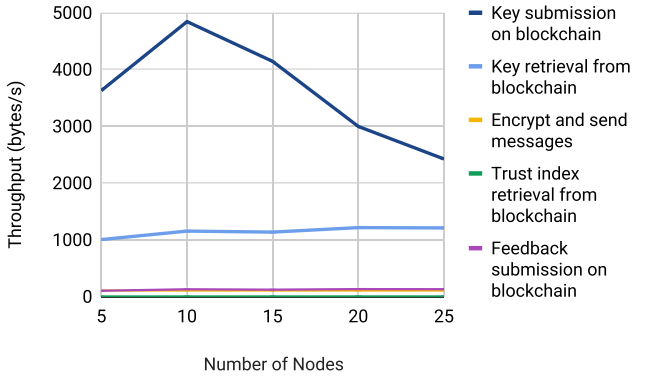


Fig. 8. Throughput when changing the number of IoT nodes.

there is no significant degradation in the performance of the throughput and access time delay in our proposed solution. We are performing the experiments in a single permission node blockchain called Multichain [20]. Multiple IoT nodes send a request and fetch the data from the blockchain. In order to check the proposed system scalability in a single node blockchain, we conducted an experiment by sending packet requests with multiple delays. The graph in Figure 8 shows the throughput as the number of nodes is increased on various steps. It shows that as we increase the number of IoT nodes, then the throughput (i.e. number of bytes transmitted per second) of the defined operations stay within a certain range.

Similarly, the graph in Figure 9 shows the throughput when the number of packets is increased on different steps. It shows that as we increase the number of packets, then the throughput (i.e. number of bytes transmitted per second) of the defined operations stay within a certain range.

The graph in Figure 10 shows the access time delay when number of IoT nodes are increased. It shows that as we increase the number of IoT nodes, then the access delay time (i.e. time taken to complete a function) of the defined operations stay within a certain range. Correspondingly, the graph in Figure 11 shows the access time delay when the number of packets is increased. It shows that as we increase

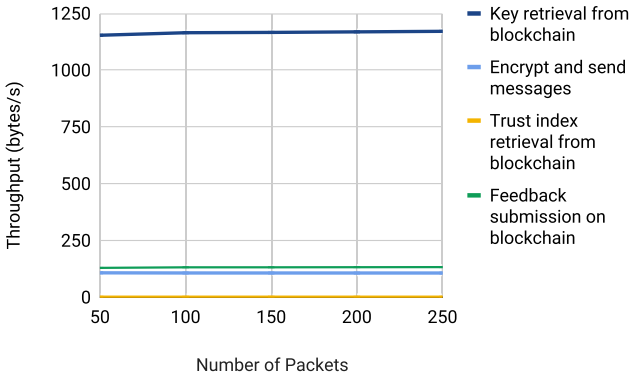


Fig. 9. Throughput when changing the number of packets.

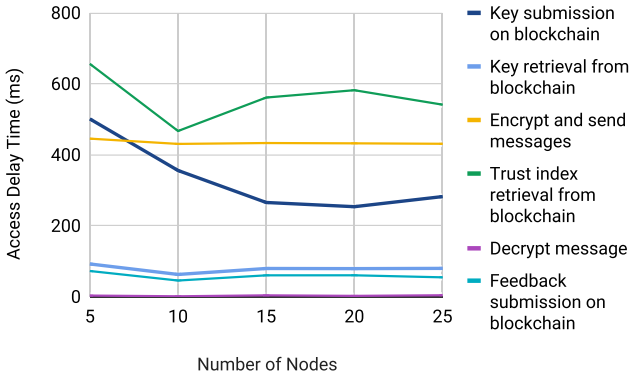


Fig. 10. Access time delay when changing the number of IoT nodes.

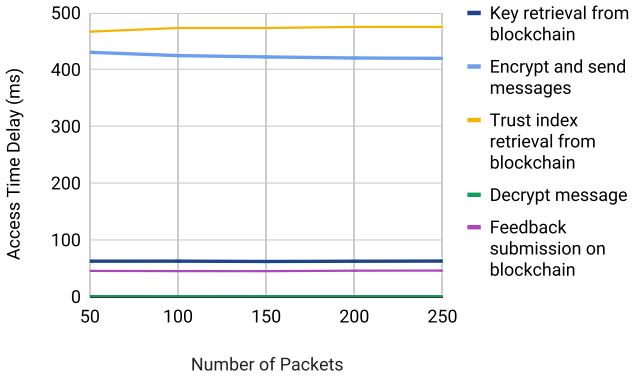


Fig. 11. Access time delay when changing the number of packets.

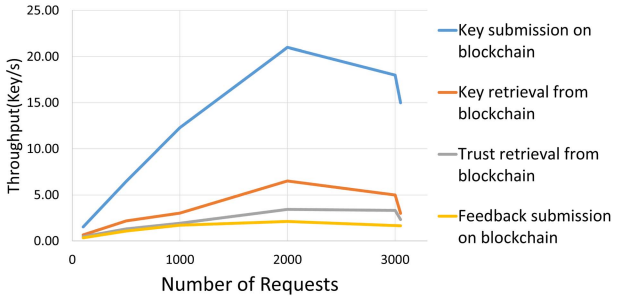


Fig. 12. Throughput when changing number of requests.

the number of packets, then the access time delay (i.e. time taken to complete a function) of the defined operations stay within a certain range.

To evaluate the scalability of the proposed solution on lager scale, we have tested the system with large number of requests to depict as if large number of nodes are operational on the network. Figure 12 shows the performance with regards to

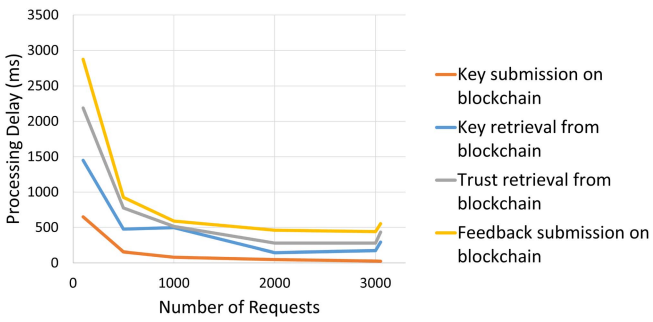


Fig. 13. Processing delay when changing number of requests.

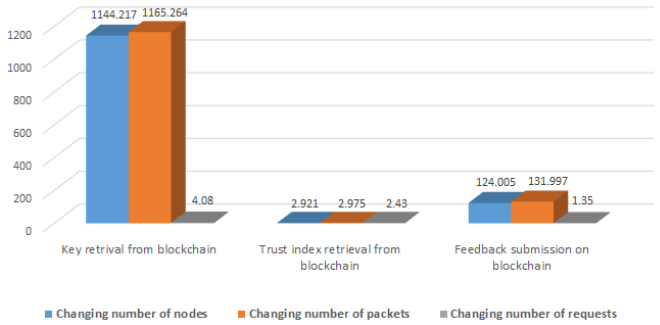


Fig. 14. Average throughput comparison of the major operations with respect to number of requests.

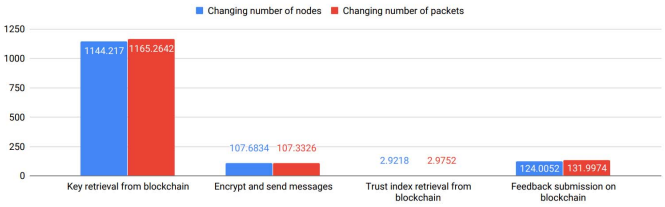


Fig. 15. Average throughput comparison of the major operations.

the throughput with increasing number of requests. It can be noted that after 3000 requests the system is failing to maintain the throughput, that is because of the computing capacity we have for this implemented testbed. Obviously, this issue can be resolved with better infrastructure facilities. Figure 13 shows the performance with regards to processing delay on changing the number of requests in various operations.

We also found out that among all the six operations, encrypting and sending the message and trust index retrieval from blockchain were the most expensive operations that took around 400 - 500 ms. Then, key submission on blockchain took around 250 - 400 ms. All the other operations took less than 100 ms and the least expensive operation was decryption of the message that took 1 - 3 ms.

Figure 14 shows a comparison of throughput for changing number of IoT nodes, packets and requests, on the three main operations. We have taken an average value of all the operations when the number of IoT nodes, packets and requests were changed systematically. The reason for not having a comparison for key submission on blockchain is because we did not capture its access delay time in our experiment and we were working on processing delay parameter for the requests.

Figure 15 shows a comparison of throughput for changing number of IoT nodes and packets of the four main operations.

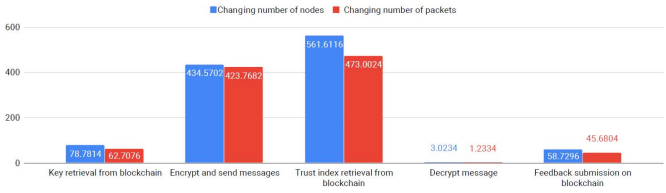


Fig. 16. Average access time delay comparison of the major operations.

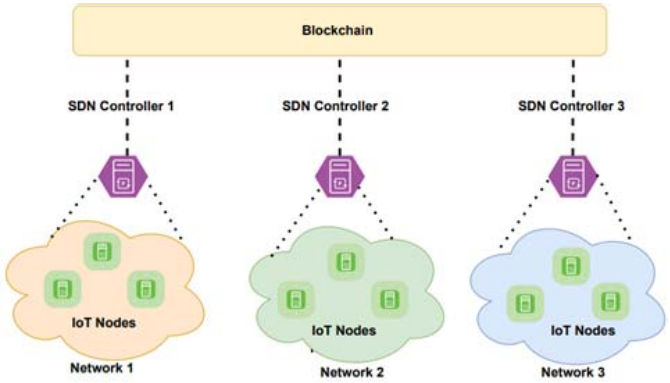


Fig. 17. New Testbed Topology with three different networks (consisting of the SDN-wise controller and IoT controller) connected to Blockchain.

We have taken an average value of all the operations when the number of IoT nodes and packets were changed systematically. The reason for not having a comparison for key submission on blockchain is because we did not capture its access delay time in our experiment. Similarly, we have no comparison for decrypt message because there was no transfer of data in this operation. The graph in Figure 16 shows a comparison of access time delay of the five main operations. The sixth operation i.e., key submission on blockchain is not compared because of unavailability of its access delay time in our experiment.

For further experimentation with the scalability assessment test on the proposed solution's multi-node blockchain, an increased number of nodes in the blockchain configuration are evaluated along with the requests load from IoT nodes. For this purpose, we performed an entirely separate set of experiments. We have created a new testbed in which three physical machines/servers are connected to the blockchain node. Each server in the new testbed consists of the SDN-wise controller and IoT controller with the Cooja simulator's help, running on the same network, as shown in Figure 17. All the three physical machines on the network are connected to the MultiChain blockchain named sdn-bc-iot. The configuration of the three deployed physical machines are mentioned in Table XVIII. During simulation in each network, we increase the number of IoT nodes with a constant 100 request per node in a network with a delay of 600 ms. The total duration time of each simulation with an increasing number of IoT nodes is given in Table XIX.

Figure 18 shows that if the number of IoT nodes in connected SDN controllers to the blockchain increased by sending a constant 100 number of requests, each with a 600 ms delay, the system's throughput decreases. Similarly, Figure 19 shows the performance concerning processing delays in various operations. We found out that encrypting and sending

TABLE XVIII
CONFIGURATION OF THE THREE DEPLOYED PHYSICAL MACHINES

Physical Machines 1, 2 and 3	
CPU (Processor)	Intel® 6th Gen Intel® Core™ i7 (6700)
Memory	16GBDDR
Chipset	Intel® H110 Chipset
Hardrive	512GB Solid State Drive SATA

TABLE XIX
SIMULATION TIME WITH INCREASING NUMBER OF IOT NODES

IoT Nodes	Simulation Time
30 Nodes (10 Nodes each Network)	50 Minute
60 Nodes (20 Nodes each Network)	117 Minute
120 Nodes (40 Nodes each network)	285 Minute
180 Nodes (60 Nodes each network)	545 minute
200 Nodes (66 Nodes each Network)	666 Minute

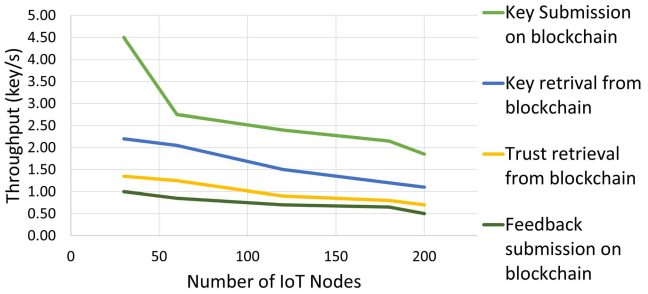


Fig. 18. Throughput with 100 requests with increasing number of IoT nodes on multi-node blockchain.

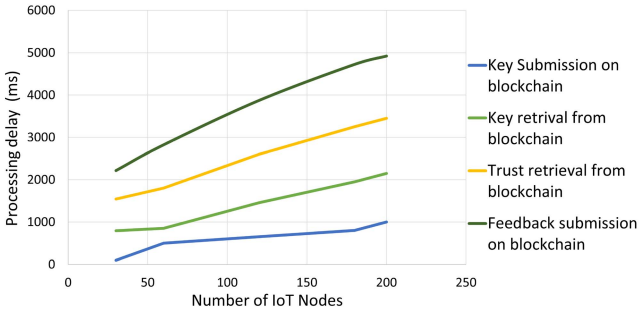


Fig. 19. Processing delay with increasing number of IoT nodes on multi-node blockchain.

the message and trust index retrieval from blockchain among all the six functions were the most expensive operations that took around 1500 – 50,000 ms. Then, the key submission on blockchain took about 99ms - 1000 ms. Table XX to XXIII shows the processing delay and the throughput values with the increasing number of IoT nodes on 1) Key submission request on blockchain 2) Key retrieval from blockchain 3) Trust index retrieval from blockchain, and 4) Feedback submission on the blockchain, respectively.

In order to minimize the random error, we obtained the reading of each experiment by performing it three times and taking its average. We also observed that there was no significant data skew in any of the readings that we captured. Based on the results, we can conclude that our solution is scalable for key and trust management in IoT networks using

TABLE XX
KEY SUBMISSION REQUEST ON BLOCKCHAIN

Number of IoT nodes	Number of Request	Throughput (requests/s)	Processing Delay(ms)
30	3000	4.50	99.11
60	6000	2.75	500.00
120	12000	2.40	652.65
180	18000	2.15	800.06
200	20000	1.85	999.84

TABLE XXI
KEY RETRIEVAL REQUEST ON BLOCKCHAIN

Number of IoT nodes	Number of Request	Throughput (requests/s)	Processing Delay(ms)
30	3000	2.20	793.27
60	6000	2.05	851.92
120	12000	1.50	1454.92
180	18000	1.20	1951.37
200	20000	1.1	2149.46

TABLE XXII
TRUST RETRIEVAL REQUEST ON BLOCKCHAIN

Number of IoT nodes	Number of Request	Throughput (requests/s)	Processing Delay(ms)
30	3000	1.35	1542.03
60	6000	1.25	1802.18
120	12000	0.90	2602.15
180	18000	0.80	3251.54
200	20000	0.7	3449.38

TABLE XXIII
FEEDBACK REQUEST ON BLOCKCHAIN

Number of IoT nodes	Number of Request	Throughput (requests/s)	Processing Delay(ms)
30	3000	1.00	2218.00
60	6000	0.85	2827.25
120	12000	0.70	3876.61
180	18000	2.13	461.47
200	20000	0.65	4727.65

SDN for network transmission and blockchain for storing the keys and trust indices of IoT devices.

VI. CONCLUSION

This paper presents a novel scalable solution for key and trust management of IoT devices in IoT networks. We proposed a framework that focuses on, 1) the registration process of the IoT devices, 2) secured and trusted communication between IoT devices, and 3) providing feedback of other IoT devices based on their communication experience. The solution is successfully set up and deployed through a virtual and simulated network of IoT devices that is centrally managed by a virtual SDN controller on a Linux machine. Along with this, a blockchain on the same Linux machine is successfully deployed using MultiChain. The implementation is effectively able to cover the registration, communication and feedback part of the proposed design i.e. when IoT devices join the IoT network, then they generate a pair of private and public key on the device itself. The IoT devices expose their public key to the SDN controller and SDN controller is able to use its private key to sign the identity of IoT device bound by its public key and publish it on the blockchain. Later, when an IoT device wants to communicate with another IoT device, then it asks the SDN controller to fetch the public key of the other IoT device and use it to encrypt and send a message. Upon receiving a message, the IoT device asks the SDN controller to fetch the IoT device's reputation from blockchain and calculate the trust index. If the trust index exceeds a certain threshold, then the IoT device decrypts the message using its private key. Finally, the IoT device rates its experience with the other IoT device and asks the SDN controller to submit feedback on the blockchain for that IoT device. Multiple sets of experiments are performed on the simulated environment by changing the number of IoT nodes and packets for the evaluation process. We found out that when we increase the number of IoT nodes and packets, then there is no major impact on the throughput and access delay times. We examined the evaluation method by submitting keys on the blockchain, retrieve keys from blockchain, encrypt and send messages, retrieve trust indices from blockchain, decrypt messages and submit feedback on the blockchain. Our evaluations proved that the proposed solution is scalable when the number of IoT nodes and requests are increased in an IoT network. Moreover, to evaluate the proposed solution's scalability on a larger scale, we experimented with an increasing number of IoT nodes to find its throughput and processing delay. We also performed separate set of experiments with multi-node blockchain to further evaluate the proposed framework. Therefore, we can conclude that our proposed solution which includes blockchain as decentralized storage for public keys and reputation history of IoT devices along with SDN for routing network traffic provides a scalable and effective solution.

The current solution can be extended in the following directions as future work:

1) Dynamic load balancing with distributed SDN-WISE controllers.
2) Scaling multiple heterogeneous IoT networks with the proposed SDN-WISE and Blockchain solution.

REFERENCES

[1] S. Moin, A. Karim, Z. Safdar, K. Safdar, E. Ahmed, and M. Imran, "Securing IoTs in distributed blockchain: Analysis, requirements and open issues," *Future Gener. Comput. Syst.*, vol. 100, pp. 325–343, Nov. 2019.
[2] B. Alotaibi, "Utilizing blockchain to overcome cyber security concerns in the Internet of Things: A review," *IEEE Sensors J.*, vol. 19, no. 23, pp. 10953–10971, Dec. 2019.
[3] D. B. Rawat and S. R. Reddy, "Software defined networking architecture, security and energy efficiency: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 325–346, 1st Quart., 2017.
[4] Y. Jararweh, M. Al-Ayyoub, A. Darabseh, E. Benkhelifa, M. Vouk, and A. Rindos, "SDIoT: A software defined based Internet of Things framework," *J. Ambient Intell. Humanized Comput.*, vol. 6, no. 4, pp. 453–461, Aug. 2015.
[5] J. Li, Z. Zhao, R. Li, and H. Zhang, "AI-based two-stage intrusion detection for software defined IoT networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2093–2102, Apr. 2019.
[6] J. Wan *et al.*, "Software-defined industrial Internet of Things in the context of industry 4.0," *IEEE Sensors J.*, vol. 16, no. 20, pp. 7373–7380, 2016.
[7] C. Aggarwal and K. Srivastava, "Securing IOT devices using SDN and edge computing," in *Proc. 2nd Int. Conf. Next Gener. Comput. Technol. (NGCT)*, Oct. 2016, pp. 877–882.

[8] O. Flauzac, C. González, A. Hachani, and F. Nolot, "SDN based architecture for IoT and improvement of the security," in *Proc. IEEE 29th Int. Conf. Adv. Inf. Netw. Appl. Workshops*, Mar. 2015, pp. 688–693.

[9] M. Tortonesi, J. Michaelis, A. Morelli, N. Suri, and M. A. Baker, "SPF: An SDN-based middleware solution to mitigate the IoT information explosion," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2016, pp. 435–442.

[10] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008, doi: 10.1145/1355734.1355746.

[11] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K.-R. Choo, "An energy-efficient SDN controller architecture for IoT networks with blockchain-based security," *IEEE Trans. Services Comput.*, vol. 13, no. 4, pp. 625–638, Jul. 2020.

[12] M. M. Rathore, A. Ahmad, A. Paul, and S. Rho, "Urban planning and building smart cities based on the Internet of Things using big data analytics," *Comput. Netw.*, vol. 101, pp. 63–80, Jun. 2016.

[13] P. Mishra, D. Puthal, M. Tiwary, and S. P. Mohanty, "Software defined IoT systems: Properties, state of the art, and future research," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 64–71, Dec. 2019.

[14] Y. Liu, Y. Kuang, Y. Xiao, and G. Xu, "SDN-based data transfer security for Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 257–268, Feb. 2018.

[15] N. Kumar and D. P. Vidyarthi, "A green routing algorithm for IoT-enabled software defined wireless sensor network," *IEEE Sensors J.*, vol. 18, no. 22, pp. 9449–9460, Nov. 2018.

[16] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for wireless sensor networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 513–521.

[17] S. Jangirala, A. K. Das, and A. V. Vasilakos, "Designing secure lightweight blockchain-enabled RFID-based authentication protocol for supply chains in 5G mobile edge computing environment," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 7081–7093, Nov. 2020.

[18] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki—A lightweight and flexible operating system for tiny networked sensors," in *Proc. 29th Annu. IEEE Int. Conf. Local Comput. Netw.*, Nov. 2004, pp. 455–462.

[19] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with COOJA," in *Proc. 31st IEEE Conf. Local Comput. Netw.*, Nov. 2006, pp. 641–648.

[20] G. Greenspan, "MultiChain private blockchain—White paper," Coin Sci., London, U.K. Accessed: Aug. 18, 2020. [Online]. Available: http://www.multichain.com/download/MultiChain-White-Paper.pdf

[21] D. E. Kouicem, A. Bouabdallah, and H. Lakhlef, "Internet of Things security: A top-down survey," *Comput. Netw.*, vol. 141, pp. 199–221, Aug. 2018.

[22] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Gener. Comput. Syst.*, vol. 82, pp. 395–411, May 2018.

[23] D. Wu *et al.*, "Towards distributed SDN: Mobility management and flow scheduling in software defined urban IoT," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1400–1418, Jun. 2020.

[24] M. B. Yassein, S. Aljawarneh, M. Al-Rousan, W. Mardini, and W. Al-Rashdan, "Combined software-defined network (SDN) and Internet of Things (IoT)," in *Proc. Int. Conf. Electr. Comput. Technol. Appl. (ICECTA)*, Nov. 2017, pp. 1–6.

[25] H. Zemrane, Y. Baddi, and A. Hasbi, "SDN-based solutions to improve IOT: Survey," in *Proc. IEEE 5th Int. Congr. Inf. Sci. Technol. (CiSt)*, Oct. 2018, pp. 588–593.

[26] A. Kuzmin, "Blockchain-based structures for a secure and operate IoT," in *Proc. Internet Things Bus. Models, Users, Netw.*, Nov. 2017, pp. 1–7.

[27] O. Novo, "Blockchain meets IoT: An architecture for scalable access management in IoT," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1184–1195, Apr. 2018.

[28] W. Viriyasitavat, L. Da Xu, Z. Bi, and A. Sapsomboon, "New blockchain-based architecture for service interoperations in Internet of Things," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 4, pp. 739–748, Aug. 2019.

[29] R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, and K. Salah, "A user authentication scheme of IoT devices using blockchain-enabled fog nodes," in *Proc. IEEE/ACS 15th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Oct. 2018, pp. 1–8.

[30] N. Tariq *et al.*, "The security of big data in fog-enabled IoT applications including blockchain: A survey," *Sensors*, vol. 19, no. 8, p. 1788, Apr. 2019.

[31] A. Singla and E. Bertino, "Blockchain-based PKI solutions for IoT," in *Proc. IEEE 4th Int. Conf. Collaboration Internet Comput. (CIC)*, Oct. 2018, pp. 9–15.

[32] J. Won, A. Singla, E. Bertino, and G. Bollella, "Decentralized public key infrastructure for Internet-of-Things," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 907–913.

[33] K. Kataoka, S. Gangwar, and P. Podili, "Trust list: Internet-wide and distributed IoT traffic management using blockchain and SDN," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, Feb. 2018, pp. 296–301.

[34] M. Nitti, R. Girau, L. Atzori, and V. Pilloni, "Trustworthiness management in the IoT: The importance of the feedback," in *Proc. 20th Conf. Innov. Clouds, Internet Netw. (ICIN)*, Mar. 2017, pp. 325–327.

[35] N. B. Truong, T.-W. Um, B. Zhou, and G. M. Lee, "Strengthening the blockchain-based Internet of value with trust," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–7.

[36] K. A. Awan, I. U. Din, A. S. Almogren, M. Guizani, A. Altameem, and S. U. Jadoon, "Robusttrust—A pro-privacy robust distributed trust management mechanism for Internet of Things," *IEEE Access*, vol. 7, pp. 62095–62106, 2019.

[37] C. Tselios, I. Politis, and S. Kotsopoulos, "Enhancing SDN security for IoT-related deployments through blockchain," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2017, pp. 303–308.

[38] M. Ma, G. Shi, and F. Li, "Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the IoT scenario," *IEEE Access*, vol. 7, pp. 34045–34059, 2019.

[39] A. Dorri, S. S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for IoT," in *Proc. 2nd Int. Conf. Internet Things Design Implement.*, Apr. 2017, pp. 173–178.

[40] S.-C. Cha, T.-Y. Tsai, W.-C. Peng, T.-C. Huang, and T.-Y. Hsu, "Privacy-aware and blockchain connected gateways for users to access legacy IoT devices," in *Proc. IEEE 6th Global Conf. Consum. Electron. (GCCE)*, Oct. 2017, pp. 1–3.

[41] B. Yu, J. Wright, S. Nepal, L. Zhu, J. Liu, and R. Ranjan, "IoTChain: Establishing trust in the Internet of Things ecosystem using blockchain," *IEEE Cloud Comput.*, vol. 5, no. 4, pp. 12–23, Jul. 2018.

[42] P. Gallo, U. Q. Nguyen, G. Barone, and P. van Hien, "DeCyMo: Decentralized cyber-physical system for monitoring and controlling industries and homes," in *Proc. IEEE 4th Int. Forum Res. Technol. Soc. Ind. (RTSI)*, Sep. 2018, pp. 1–4.

[43] P. K. Sharma, S. Singh, Y.-S. Jeong, and J. H. Park, "Distblocknet: A distributed blockchains-based secure sdn architecture for iot networks," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 78–85, Sep. 2017.

[44] L. Dykcik, L. Chuat, P. Szalachowski, and A. Perrig, "BlockPKI: An automated, resilient, and transparent public-key infrastructure," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2018, pp. 105–114.

[45] A. Durand, P. Gremaud, and J. Pasquier, "Decentralized Web of trust and authentication for the Internet of Things," in *Proc. 7th Int. Conf. Internet Things*, Oct. 2017, pp. 1–2.

[46] H. Tewari, A. Hughes, S. Weber, and T. Barry, "X509Cloud—Framework for a ubiquitous PKI," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2017, pp. 225–230.

[47] J. Chen, S. Yao, Q. Yuan, K. He, S. Ji, and R. Du, "Certchain: Public and efficient certificate audit based on blockchain for tls connections," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 2060–2068.

[48] D. S. V. Madala, M. P. Jhanwar, and A. Chattopadhyay, "Certificate transparency using blockchain," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2018, pp. 71–80.

[49] L. Mendiboure, M. Chalouf, and F. Krief, "Towards a blockchain-based SD-IoV for applications authentication and trust management," in *Proc. Internet Vehicles. Technol. Services Towards Smart City*, Nov. 2018, pp. 265–277.

[50] J. Yuan and X. Li, "A reliable and lightweight trust computing mechanism for IoT edge devices based on multi-source feedback information fusion," *IEEE Access*, vol. 6, pp. 23626–23638, 2018.

[51] S. Namal, H. Gamaarachchi, G. MyoungLee, and T.-W. Um, "Autonomic trust management in cloud-based and highly dynamic IoT applications," in *Proc. ITU Kaleidoscope, Trust Inf. Soc.*, Dec. 2015, pp. 1–8.

[52] R. Di Pietro, X. Salleras, M. Signorini, and E. Waisbard, "A blockchain-based trust system for the Internet of Things," in *Proc. 23rd ACM Symp. Access Control Models Technol.*, Jun. 2018, pp. 77–83.

[53] F. Kandah, B. Huber, A. Skjellum, and A. Altarawneh, "A blockchain-based trust management approach for connected autonomous vehicles in smart cities," in *Proc. IEEE 9th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2019, pp. 0544–0549.

[54] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-based secure and trustworthy Internet of Things in SDN-enabled 5G-VANETs," *IEEE Access*, vol. 7, pp. 56656–56666, 2019.
[55] Z. Lu, Q. Wang, G. Qu, and Z. Liu, "BARS: A blockchain-based anonymous reputation system for trust management in VANETs," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./12th IEEE Int. Conf. Big Data Sci. Eng.*, Aug. 2018, pp. 98–103.
[56] I. Tomic and J. A. McCann, "A survey of potential security issues in existing wireless sensor network protocols," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1910–1923, Dec. 2017.
[57] S. Cirani, M. Picone, P. Gonizzi, L. Veltri, and G. Ferrari, "IoT-OAS: An oauth-based authorization service architecture for secure services in iot scenarios," *IEEE Sensors J.*, vol. 15, no. 2, pp. 1224–1234, Feb. 2015.
[58] SDN-WISE. *SDN-Wise, the Stateful Software Defined Networking Solution for the Internet of Things*. Accessed: Mar. 2020. [Online]. Available: https://sdnwiselab.github.io/

Sufian Hameed received the Ph.D. degree in networks and information security from the University of Göttingen, Germany. He works as an Assistant Professor with the Department of Computer Science, National University of Computer and Emerging Sciences, Pakistan. He also leads the IT Security Labs, NUCES. The research lab studies and teaches security problems and solutions for different types of information and communication paradigms. His research area includes network security, web security, mobile security, and secure architectures and protocols for cloud and Internet of Things (IoT).

Syed Attique Shah received the Ph.D. degree from the Institute of Informatics, Istanbul Technical University, Istanbul, Turkey. During his Ph.D. degree, he was a Visiting Scholar with the National Chiao Tung University, Taiwan, the University of Tokyo, Japan, and the Tallinn University of Technology, Estonia, where he completed the major content of his thesis. He currently works as an Assistant Professor and the Chairperson with the Department of Computer Science, BUITEMS, Quetta, Pakistan. His research interests include big data analytics, information management, and the Internet of Things.

Qazi Sarmad Saeed received the M.S. degree in computer science from the National University of Computer and Emerging Sciences, Karachi, Pakistan. His areas of interests include information management, network security, cloud computing, and the Internet of Things (IoT).

Shahbaz Siddiqui received the M.S. degree in telecommunication from Hamdard University, Karachi, Pakistan. He is currently pursuing the Ph.D. degree in computer sciences with the National University of Computer and Emerging Sciences, Karachi. He currently works as an Assistant Professor with the Department of Computer Science, National University of Computer and Emerging Sciences. His research interests include the Internet of Things and blockchain

Ihsan Ali (Senior Member, IEEE) received the M.S. degree in computer system engineering from the GIK Institute in 2008. He is currently pursuing the Ph.D. degree with the University of Malaya, Kuala Lumpur, Malaysia. He is also an active Research Associate with the Center for Mobile Cloud Computing Research (C4MCCR), Faculty of Computer Science and Information Technology, University of Malaya. He is a Research Associate with the Faculty of Computer Science and Information Technology, University of Malaya. He has published more than 40 high-impact research journal papers, including a highly reputable *IEEE Communication Magazine*. He has been actively involved in research and teaching activities for the last ten years in different countries, including Saudi Arabia, USA, Pakistan, and Malaysia. His research interests include wireless sensor networks, robotics in WSNs, sensor cloud, fog computing, Internet of Things (IoT), and ML/DL in wireless sensor networks. He is also an active Reviewer of *Computers and Electrical Engineering*, *KSII Transactions on Internet and Information Systems*, *Mobile Networks and Applications*, *International Journal of Distributed Sensor Networks*, *Journal of Advanced Transportation*, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *Computer Networks*, IEEE ACCESS, *Wireless Communications and Mobile Computing*, and *IEEE Communication Magazine*.

Anton Vedeshin received the M.Sc. degree in computer science and the Ph.D. degree in cloud computing and cybersecurity from the Tallinn University of Technology (TTÜ). He has developed software since the age of 11. He is the CTO and the Co-Founder of 3D Control Systems, Inc., working in the advanced manufacturing industry and deploying secured automated manufacturing solutions for F500 enterprises. He founded his first software development company at the age of 20 with over 25 employees. He developed multiple security and e-government solutions for the Estonian Government and the EU Commission and worked in the Mitsubishi Electric Car Data Mining Project. He designed numerous blockchain solutions based on several platforms (Ethereum, Hyperledger Fabric, and Corda R3). He teaches cloud computing at TTÜ and data security at the Estonian Entrepreneurship University of Applied Sciences (EUAS). He is the author of numerous cybersecurity and digital manufacturing scientific papers (IEEE, ACM, and Springer) and patents. He conducted reviews of numerous articles on cloud computing for IEEE ACCESS. His research interests include cloud computing, cybersecurity, 3D printing, blockchain, machine learning, and AI.

Dirk Draheim received the Ph.D. degree from Freie Universität Berlin and the Habilitation degree from the Universität Mannheim, Germany. He is currently a Full Professor of Information Systems and the Head of the Information Systems Group, Tallinn University of Technology, Estonia. The information systems group conducts research in large and ultra-large-scale IT systems. He is also an initiator and a leader of numerous digital transformation initiatives.