Computer Communications 198 (2023) 1–31

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Towards SDN-based smart contract solution for IoT access control

Mizna Khalid [a], Sufian Hameed [a,*], Abdul Qadir [a], Syed Attique Shah [b], Dirk Draheim [c]

[a] Department of Computer Science, NUCES, Karachi, 75030, Pakistan
[b] School of Computing and Digital Technology, Birmingham City University, Millennium Point, B4 7XG, United Kingdom
[c] Information Systems Group, Tallinn University of Technology, Akadeemia Tee 15a, 12618 Tallinn, Estonia

ARTICLE INFO

Keywords:
Access control
Blockchain
Internet of Things
Smart contract
Software-defined Networking

ABSTRACT

Access control is essential for the IoT environment to ensure that only approved and trusted parties are able to configure devices, access sensor information, and command actuators to execute activities. The IoT ecosystem is subject to various access control complications due to the limited latency between IoT devices and the Internet, low energy requirements of IoT devices, the distributed framework, ad-hoc networks, and an exceptionally large number of heterogeneous IoT devices that need to be managed. The motivation for this proposed work is to resolve the incurring challenges of IoT associated with management and access control security. Each IoT domain implementation has particular features and needs separate access control policies to be considered in order to design a secure solution. This research work aims to resolve the intricacy of policies management, forged policies, dissemination, tracking of access control policies, automation, and central management of IoT nodes and provides a trackable and auditable access control policy management system that prevents forged policy dissemination by applying Software Defined Network (SDN) and blockchain technology in an IoT environment. Integration of SDN and blockchain provides a robust solution for IoT environment security. Recently, smart contracts have become one of blockchain technology's most promising applications. The integration of smart contracts with blockchain technology provides the capability of designing tamper-proof and independently verifiable policies. In this paper, we propose a novel, scalable solution for implementing immutable, verifiable, adaptive, and automated access control policies for IoT devices together with a successful proof of concept that demonstrates the scalability of the proposed solution. The performance of the proposed solution is evaluated in terms of throughput and resource access delay between the blockchain component and the controller as well as from node to node. The number of nodes in the IoT network and the number of resource access requests were independently and systematically increased during the evaluations. The results illustrate that the resource access delay and throughput were affected neither linearly nor exponentially; hence, the proposed solution shows no significant degradation in performance with an increase in the number of nodes and/or requests.

1. Introduction

The evolving IoT paradigm comprises of heterogeneous, intelligent devices (smart objects) that operate pervasively in a highly-distributed manner under the restriction of a series of today's network- and device-specific resource constraints. The rapid increase in the usage of IoT devices in the last decade has stretched problems related to accessibility, performance, safety, and scalability. Unauthorized access is one of IoT's primary challenges [1,2], due to low energy requirements and ad-hoc network scenarios. If access control mechanisms are disabled, devices can expose sensitive personal information of end users. Access control is needed to ensure that only approved trusted and authorized parties may upgrade system software, access sensor details, or order actuators to execute an activity. The devices must be authenticated for privileged access to services and resources. Because of various heterogeneous fundamental architectures and systems that supports IoT technology, the need for authentication mechanisms for IoT becomes essential. Mechanisms for access control and authentication guarantee confidentiality so that data can only be accessed by approved users. Traditionally, it was achieved by strict physical, administrative, and technical controls. Mostly access control attackers evade or bypass access control mechanisms and steal the information or modify the information so that it does not remain accurate.

Access control policies can be implemented in the IoT environment either through a centralized approach or a distributed approach [3]. In a centralized strategy, the entire access control logic is outsourced to a key entity (gateway) that is responsible for filtering access applications based on their permission policies. Here, smart devices only play the role of data sources. In a distributed approach, on the other hand, the complete access control logic is integrated into the end systems.

* Corresponding author.
E-mail address: sufian.hameed@nu.edu.pk (S. Hameed).

https://doi.org/10.1016/j.comcom.2022.11.007
Received 8 February 2021; Received in revised form 27 March 2022; Accepted 8 November 2022
Available online 17 November 2022

Unfortunately, the restricted computational capabilities and processing energy in combination with heterogeneity constraints of IoT devices, are rising challenges for IoT devices to manage their access control [4].

In order to tackle management challenges, several resource management systems have been given significant attention in the IoT environment. The software defined network (SDN) paradigm offers an attractive solution for managing IoT resources. SDN brought us the concepts of remote and continuous programming of network components as well as programmable network monitoring; this way, it helps to mitigate some of the most important network issues. It proposes a new architecture that extracts the control features from packet forwarding hardware,i.e., data planes to external software controllers, i.e., control planes. SDN has been suggested to tackle the difficulties faced by the standard and traditional network control paradigm. It can provide the network programmability by decoupling control planes and data planes. SDN architecture supports a number of API's for implementing popular network services, comprising security (authentication and access control), multi-cast routing, bandwidth management, traffic monitoring and engineering, quality of service, optimization of processors, storage and energy utilization and everything related to policy management, customized to business requirements [5].

The scientists recognized and discussed IoT management and safety concerns in [6] and proposed SDN-based guidelines for enabling IoT network security services. Their proposed IoT system based on SDN comprises of three primary parts: IoT controller, security controller focused on SDN and sensor openflow switch(s). The SDN controller communicates with application security services in order to provide: access control, trust, privacy, key management, IoT network-wide authentication of service access and mitigation of security attacks that are deployed as modules to enable IoT network security. Despite a great effort by the authors, to develop a comprehensive framework no implementation or evaluations of the proposed framework are presented. In order to provide better access management in IoT and fix security problems, the authors in [7] highlighted the need to provide automated IoT authentication systems where scalability is essential along with platform heterogeneity and versatility. The authors suggested a smart contract-based access control framework to obtain distributed trustworthy access control for IoT applications, consisting of: multiple access control contracts (ACCs), one judge contract (JC) and one register contract (RC), to obtain distributed and trustworthy access control for IoT systems. The model presented numerous features of using smart contracts (ACC, JC and RC) for IoT environment. These features involve: updation and registration for the misbehavior-judging method (add, update and delete an ACC policy), and access policies for a pair of subject-objects. Each ACC offers one access control technique for a subject-object set that performs both static access correct validation based on predefined access control strategies and dynamic access validation by inspecting the subject's actions. Although this paper uses smart contracts to provide a dynamic and expressive access control methods, the fact that each Access Control Contract (ACC) corresponds to one access control method for a subject-object pair means that the proposed framework relies on the traditional Access Control List (ACL) approach to manage access control. However, management of contracts would become complex when the number of subjects and objects increases.

Since its introduction as the cryptocurrency Bitcoin, blockchain technology [8] has received massive attention in scholarly [9,10] and practical industrial applications [11,12]. The features and capabilities of blockchain technology have a huge potential to address IoT security problems [13]. Blockchain and smart contracts can be exploited to ensure that access to information is privacy-protected and traceable as well as time-stamped and tamper-proof; i.e., they can help against unwanted human intervention. The essential characteristics of each blockchain-based IoT application is the integration of smart contracts with the underlying distributed ledger plus IoT infrastructure. Smart contracts go beyond the original purpose of blockchain technology as a cryptocurrency; they open blockchain technology for an endless range of applications and domains such as digital identity management (including MyData[1] and self data), healthcare, logistics, B2B (business-to-business), B2C (business-to-customer), business continuity management etc. [14].

The motivation for this work is to mitigate the existing predicaments of IoT related to access control policy management and security. In designing the security solution to individual features of each IoT domain, due consideration is given to employing specific access control policies satisfying the concern. Table 1 provides an overview of security needs together with their respective implications in different IoT domain environments. This paper objectively targets the resolution of complexity in policies management, forgery, dissemination, and tracking of access control and central management of IoT nodes. The proposed work aims at better, automated access control for IoT against unauthorized access through teaming up SDN and blockchain technology.

The combination of SDN with blockchain allows the network to be managed centrally along with configurable functionalities [25]. The control and configurable nature of SDN make it applicable to various network topologies [26]. Whereas, by employing blockchain technology in the IoT environment, advancement in the automation of business processes can be observed without the necessity for the implementation of complex centralized network infrastructure. Building trust between devices and users diminishes the risk of deception, reduces costs, and eliminates reliance on third parties. Blockchain-based IoT solutions are perfect for simplifying business automation, considerable cost-saving, and improving user experience [18].

In our solution, blockchain-based smart contracts are used to define the rules and penalties related to an agreement and beyond, as they automatically enforce obligations and establish the immutability of the contracts. The proposed work aims at better, automated access control for IoT against unauthorized access through teaming up SDN and blockchain technology. The adaption of SDN for the IoT environment is termed as SD-IoT (Software-Defined Internet of Things). The role of the integrated SDN controller (SDN-WISE controller) will be to provide structure and centrality to the smart contracts and devices. In our proposed work, we integrated SDN with blockchain smart contracts in order to achieve manageable, automated, verifiable and immutable access control for IoT environment. SDN controller encourages the management of contracts while blockchain provides immutability feature for access control policies. Moreover, to analyze and benchmark the performance, in terms of throughput and resource access delay we deployed prototype testbed where the number of nodes were increased systematically.

The major contributions of this paper can be summarized as follows:

(a) A novel scalable solution for implementing immutable, verifiable, adaptive and automated access control policies for IoT networking is proposed; together with a successful proof of concept that demonstrates the scalability of the proposed solution.
(b) The integration of an IoT network along with blockchain and SDN technologies is effectively exhibited through simulation deployment that is able to store access control policies of IoT devices on the blockchain and manage the network traffic efficiently through SDN. We observed and benchmarked the effectiveness of using Smart Contracts with Software-Defined Internet of Things (SD-IoT) in the IoT environment for the access control solution.
(c) The performance of the proposed solution is evaluated in terms of throughput and resource access delay between blockchain to controller and from node to node. The number of nodes in the IoT network and the number of resource access requests were independently increased for evaluations. The results illustrate that the number of nodes had a dominant impact as compared to the

[1] https://mydata.org/declaration/

Table 1
IoT-domain-based comparison for various objectives covered by selected studies.

Studies	Domain	Reliability & Availability	Confidentiality & Integrity	Access control
[15,16]	Smart Grid	Highly Critical	Required	Required
[17,18]	Smart City	Required	Highly Required	Highly Required
[19,20]	Smart Home	Highly Required	Highly Required	Highly Required
[21,22]	Health Care	Highly Required	Highly Required	Highly Critical
[23,24]	Industry 4.0	Highly Critical	Moderate Required	Highly Critical

number of requests. When nodes are kept fixed the throughput does not change significantly and the observations for different numbers of resource access requests produced minor changes in throughput and resource access delay. Conversely, the increase in the number of nodes caused slight increase in the delay for resource access. Our evaluation results are quite promising and demonstrates the effectiveness of the proposed architecture.

The rest of the paper is structured as follows. Section 2 presents to the reader the prescribed background information on both blockchain technology and smart contracts, focusing particularly on software-defined networks along with SD-IoT and access control in IoT. Section 3 investigates existing work carried out with respect to access control in IoT using SDN and blockchain technology. Section 4 thoroughly describes the architecture of the proposed solution. In Section 5, the prototype implementation is described. Section 6 focuses on testbed implementation and its evaluations. This section describes various scenarios for our experiments and presents their results. Finally, Section 7 presents our conclusions about the research effort.

2. Background

2.1. Blockchain technology and smart contract

Blockchain technology [8] has recently gained a considerable place in the research world [9,10] as a rising and evolving technology for instant verification of transactions and dealings between business trades, private and public industries and organizations [11,12]. A Blockchain comprises a data structure that is unchangeable and distributed, and can be replicated and shared between all network members. The peer-to-peer nature of blockchain provides the system that operates on top of it with scalability and fault tolerance. A Blockchain enables verifiable interaction in a distributed peer-to-peer network where potentially untrustworthy members communicate with each other. Blockchain technology as a primary archive of a certain distributed scheme that enables approved nodes to immediately monitor and check information produced by IoT devices once recorded, irrespective of their quantity or the total amount of sources [27,28].

With the help of smart contracts [29], blockchain technology can be expected to play an essential role in managing, monitoring and (most importantly) in maintaining information security of IoT smart devices. Smart contracts are programmable applications that are stored in the blockchain for managing transactions under specific conditions and terms. In other words, smart contracts are the digital equivalent of traditional financial agreements between different entities involved [30]. They take the role of performing transactions in blockchain networks in a predetermined fashion, agreed upon by contracting parties. For IoT devices, blockchain smart contracts provide the capability of decentralized authentication regulations and logics backed by single-party and multi-party verification. A smart contract offers various application binary interfaces (ABIs) that can be operated by any peer in the blockchain scheme. In addition to ABIs, there are also data in a smart contract, which is considered the contract state. Each smart contract is associated with an address that allows any peer in the blockchain system to execute their ABIs and change their state if the change is allowed. All peers in the system will execute the ABIs; so long as the computing capacity of any peer in the whole system is less than half. Hence, no peer in the system can intentionally execute the ABI in the wrong way. As a result, peers cannot manipulate smart contract features [7]. Similarly, in comparison with traditional protocols for authentication, authorization and verification such as OpenID, Role-Based Access Management (RBAC), Attribute-based access control (ABAC), OAuth, OMA DM, SAML and LWM2M, we have that blockchain-based smart contracts are capable of providing a less complex, more efficient, immutable and verifiable set of access rules for linked IoT smart devices [31]. These protocols are currently widely used for authenticating, authorizing and managing IoT devices. Achieving data security and privacy can be managed by the feature of smart contracts that sets the rules and policies for access control, circumstances and allows the time for ownership control and access to information by authorized individuals or groups of user devices during rest or transit. Blockchain makes it possible to verify its attributes. Transactions and security policies based on blockchain are easy to audit. Because of this and other characteristics, blockchain technology can offer a significant part in monitoring uncertainty of sources as well as managing and addressing crisis situations. IoT security, identity and access management challenges such as IP spoofing can also be addressed by blockchain-based systems [32].

2.2. Software defined networking

Software-defined networking (SDN) [33] is a young [34], innovative networking paradigm that tackles the complexities and difficulties of the traditional network architecture by dividing all control and management activities from the underlying system components and putting them into a software layer, a middleware layer. The introduction, growth and creation of SDN and the idea of programmable networks, has recently regained considerable popularity. SDN, an innovative paradigm enables network operators to handle and program their network more flexible and resolves the restriction of legacy networks. SDN simplifies network management by separating the control plane (making data forwarding decisions) from the data plane (forwarding elements) as well as making the network flexible so that it can be deployed and programmed automatically. By dynamically programming and reorganization network environments from the central SDN controller, SDN simplifies network setup. The controller is not restricted to communicate with any function or process to make adequate choices on traffic management [6,35]. Defining the network's security policies with the software-controlled nature of the SDN, network management becomes simple. In SDN, only the controller must be implemented by a network administrator, which is then replicated across the data forwarding devices of the network. The primary objective is enabling the software engineers to depend on resources of networks as easily as they do on storage and computer resources [33].

Efficient use of resources is a significant task confronting potential networks; this is particularly the situation in multi-hop wireless ad-hoc networks since the accessibility of wireless power is fundamentally restricted. It can be because of a variety of conditions such as using the compounded shared physical media, impairment of wireless channels or due to lack of controlled infrastructure. Although these self-organized wireless ad-hoc networks are used in an overburdened infrastructure to complement or fill the gaps, their absence of committed resources and changing technology makes it hard to share [36]. Networks and nodes major heterogeneous features (e.g. physical media, structure, stabilization, buffer size, energy constraints, and mobility)

are additional significant factors which contribute to routing and resource distribution considerations. SDN ensures the ability of efficient and simple implementation and management of network services. SDN methods, such as OpenFlow, mainly target networks depending on services. SDN architecture encourages the concept of a control system that is centralized and incompatible with the amount of de-centralization, interference, and interruption present in a network infrastructure.

2.3. SD-IoT

There is exponential growth in the connectivity of heterogeneous gadgets, to the internet in changing the Internet of Things (IoT) technology. Securing such complicated heterogeneous networks and their varied methods of access are a true task that leads to safety danger [37]. SDN paradigm provides an appealing alternative for managing recently under focus IoT services [38]. It has the ability to track traffic smartly and utilize network resources, which are not frequently used [39]. SDN presents separation of concerning issues regarding the control plane (takes traffic handling decision) and the data plane (real traffic transmission processes to desire locations). The decoupling promotes the abstraction of lower-level network functionality into higher-level facilities and thus network administration tasks become easy to manage. This will considerably improve the capacity of the network and therefore planning for IoT's data attack will be much easier for systems . This will eliminate inefficiencies to process IoT-generated information effectively without putting a major burden on the network, particularly on the Wi-Fi network [40].

IoT and SDN integration simplifies the process of acquiring information, analyzing information, making decisions and implementing actions. Implementation of SDN in IoT provides monitoring and management of network assets and ensures access management depending on user, organization, device, and implementation that ultimately permits users and even devices to exchange data capacity. IoT networks can gain benefit from the development of SDWN (Software Defined Wireless Networking) to enhance the capacity to control networks. With SDWN, IoT networks based on demand can become more scalable [40].

2.4. Access control in IoT networking

IoT devices need to be allowed to access network services when they join and register in a new network. In that case, whenever a malicious node manages to become a part of the network, it can execute activities that are malicious and can either lead IoT facilities to disrupt and alter sensitive information in the network. IoT nodes should join a network with authentication algorithms and security policies to deter IoT nodes from executing malicious activities [6]. The IoT system or systems that can be manipulated can be taken over by malicious individuals or groups within the network. An IoT smart device, therefore, must be resilient to manipulation and tampering and meet all security demands [19]. Therefore, there is a desire for a solid security management alternative that guarantees resilience to such manipulation. IoT atmosphere requires understanding when and by whom to access their facilities. This will guarantee a greater standard of security is maintained. It is a difficult job to maintain check and balance of services in the IoT environment. A centralized perspective of the IoT network with SDN management structure can assist in the logging of IoT network operations [6]. Only users who are verified through access control can utilize services and applications, such as device or sensor information or any data file. All contemporary working technologies restrict user-based access to the file scheme. Access control is required in the IoT framework to ensure that only trusted users are able to update system applications, gain access to sensor information and request actuators to conduct any specific activity.

The IoT ecosystem requires different access control systems due to low power consumption and limited latency between the connected devices and the network, distributed system configuration, heterogeneous computer networks and the ability to connect an extremely wide range of devices. Hence, conventional authorization and verification designs such as Access Control List (ACL), Role-Based Access Control (RBAC), Attribute-Based Access Control (ABAC) and Capability-Based schemes (CBAC) need to be thoroughly evaluated before they are applied to the IoTs. To secure sensitive device-generated IoT information from cyber-attacks, current alternatives generally apply cryptographic methods only to approved users by disclosing data decryption keys. These alternatives, however, are vulnerable to a variety of cybersecurity attacks but allow authorized entities to perform activities and utilize resources or information from connected devices. Moreover, cryptographic methods for key distribution and information management cause computation overhead at the device node. For this reason, IoT requires an automated and self-contained access control without relying on the computational capabilities of the device or dealing with the issue of password fatigue in which applicants are required to manage passwords for different services and applications [41].

2.5. Major security challenges to consider in IoT networks

Some of the major security challenges and requirements that should be considered are as follows:

- Trustworthy relationship among devices and management of devices, since IoT nodes are heterogeneous, having different computational capabilities, security requirements, and belonging to different vendors.
- Unauthorized access of resources must be prevented.
- Unauthorized tampering of information must be prevented, and integrity of information must be ensured.
- Confidentiality and privacy of information.
- Reducing computational processing related to access control of resources.

During these security-related scenarios, SDN and blockchain integration complement one another. The following are some of the most critical security challenges and requirements covered in this paper:

1. **Administration of access**
   Unauthorized access of resources is prevented by setting predefined policies and penalties if any misbehavior occurs. Access regulations and policies must be enforced by the IoT resource owners in order to prevent unauthorized use and to secure the resources and sensitive information.
2. **Bonding limits of nodes**
   Since IoT nodes are heterogeneous, with various computing capabilities, security needs, and belonging to different manufacturers, trustworthy relationships between devices and device management are established through controllers and access control modules.
3. **Security and protection towards malicious node attacks**
   As nodes gets authorized through a consensus mechanism (proof-of-authority [42,43]) to attain permission for entering the network and generating new blocks, it becomes difficult for nodes to turn malicious.
4. **Significance of accepted policies**
   Once the policies are deployed on the blockchain by the node, tampering is not possible even if the nodes become malicious. If nodes become malicious their trust reputation will be denigrated eventually. For trust evaluation, several existing solutions can be considered and can be executed according to the domain requirements.
5. **Ensuing Benefit of blockchain**
   Unauthorized tampering of information is prevented, and integrity of information is ensured through blockchain. Once the access policies are deployed, they become immutable.

3. Related work

In the last few years there have been extensive efforts in defining secure IoT access control. This section discusses different IoT access control mechanisms, as proposed by the various researchers in the existing literature. Table 2 briefly outlines the intended approaches for access control and the shortcomings observed in the respective research, wherein the application of the latest access control is debatable due restricted IoT environment. The table also presents a comparative study approach where the investigated solutions are significantly embracing the objective of the research.

Traditional security access control systems are unable to cope efficiently with multidimensional city problems such as traffic jams, lengthy queues, waiting periods, etc. Smart cities are built on autonomous and distributed infrastructure, which includes autonomous data processing and control systems, heterogeneous network architecture involving millions of sources of information. Smart cities must innovate to track and manage their populations, vehicles, buildings and tourist sites with better and smarter access control systems [17]. It is possible to use intelligent and connected devices in a diverse range of city conditions. Connecting them to the main operating center for linked access control provides considerably efficient city management. However, being aware of the cybersecurity hazards is crucial as there are multiple entry points in such a large network of smart devices [44]. Access control, in health care, has essential significance. Protecting data against unlawful access and consequently misuse or legal liability is crucial. Privacy is a major challenge when it is related to the patient's medical information that needs to be confidential. In [21], the scientists evaluated and researched the design and execution of an access control system for the healthcare domain, in particular within EMR schemes. Improving the design and use of the access control system can decrease some of the obstacles to EMR inclusion owing to which care and support for patients can be enhanced.

Unlike traditional access management techniques, in [45] the researchers recommended an innovative access control framework for IoT environment called "SmartOrBAC". Based on the OrBAC model, which excels in a centralized system, but has some observable limitations, which include addressing collaboration between Organizations and sub-organizations, and the rendition of the security policy into the access control mechanism. SmartOrBAC, an extension of OrBAC, proposes to address these limitations. It uses web services to ensure secure collaboration between different organizations, and also emphasizes on using RESTFUL API for exchanges between the organizations, benefiting from its light mechanism. SmartOrBAC also offers an efficient access control solution for low power and energy-constrained collaborative entities such as IoT. Intended for a decentralized safety structure, the researchers suggested a new framework for smart grids in [15], incorporating privacy and access control at the same time. The security and privacy issue are an important and crucial issue associated with the smart grid, not only because of potential terrorist attacks but also because of conceivable manipulation of different devices by clients and building officials. Directed to its significance usage, smart grid privacy has been widely analyzed. Home Area Network (HAN), Building Area Network (BAN), and Neighborhood Area Network (NAN) information gateway smart meters are aggregated using homomorphic encryption. This system is decentralized, composed of different main distribution centers (KDC), avoiding any single point of failure. The access control scheme is focused on primitive cryptographic so-called attribute-based encryption (ABE), granting restricted access to data consumers such as monitoring teams, analytical maintenance teams, engineers, environmentalists, study organizations, policymakers, management organizations, etc. Users have attributes and these attributes are dependent on access strategies. Once the broadcast message is encrypted then it can only be decrypted by authorized users (with valid access policy). In this safety structure, malicious, unauthorized and prohibited user nodes can be removed with comparatively small overhead. In [16], the researchers present the idea of Smart-grid Operation-based Access Control (SOAC), innovative access control for smart-grid computer network aimed at increasing the power-grid reliability and security. The SOAC is configured to expand the access control capabilities of traditional role-based access control for the network originally intended for generalized network security, non-electrical utility operations. SOAC offers the idea of a large network from local micro-grid domains to integrated local micro-grid domains and regional micro-grid domains as expected for potential smart grid expansions. For modernizing the power grid systems, SOAC offers a safety architecture that fulfills the fundamental security demand for safe open access to the smart grid.

The requirement for a secure and reliable third party to manage access control reasoning is also counterproductive to the safety of the user. Designing immutable automated access control with user-driven strategy and privacy-preserving awareness in the IoT environment is therefore of great importance. The blockchain platform, to tackle this IoT environment limitation represents a groundbreaking approach for developing a new generation of open and clear access control solution that provides end user more active and secure privacy [46]. Several initial research in this area started to investigate this emerging field and investigate the problem of access management in IoT utilizing blockchain related solutions. A survey is conducted in [31] to study main security problems in the Internet of Things (IoT) domain. The authors have analyzed classification of IoT security threats. More specifically, they discussed how to address security standards in IoT using the blockchain technology.

Centralized traditional access control solutions tend to be less IoT tailored [62]. In [63], the authors highlighted the need of providing automated authentication systems for scalable, heterogeneous and versatile environment of IoTs. The researchers are proposing a blockchain-based decentralized solution for smart homes [49] to tackle these problems, which dynamically collects device signatures to recognize all devices and their customers. In [64], the authors provided detailed description of how blockchain and smart contracts work and highlight the ways of using blockchains and IoT together for immutable communication. Blockchain and IoT combination can be very powerful. Blockchain provides resilient, genuinely dispersed peer-to-peer technologies and the capacity to communicate with peers ensuring trust and audibility. Smart contracts enable complicated multi-step procedures to be automated. Integrating blockchain and smart contract into the IoT domain can result in significant changes across multiple sectors, generating fresh business models, and can be used to reconsider how current technologies and procedures are being applied. Another important work is proposed in [50] which utilizes the consistency of blockchain technology to handle access control issues in IoT. Authors in this paper presented a new access control environment that supports the scalable access control of trillions of IoT devices. The key aspects of this research is the idea for an implementation that addresses the question of integrating low-storage IoT devices in the distributed infrastructure, in addition to the broad scale of the blockchain. The approach consists of creating new nodes called a management hub that demands access control details on request of the IoT devices from the blockchain. However, the description and specification of the access control policies have not been provided. Similarly, in [51] authors conducted a research and proposed auditable and manageable access control framework for decentralized online social networks implementing blockchain technology for the defining privacy policies. The authors compared the results and proved the efficiency in terms of gas cost of ACL-based access control over the Attribute-based access control (ABAC). In contrast to previous work, authors in [52] describe an interesting approach for managing and implementing Attribute-based Access Control (ABAC) policies expressed in eXtensible Access Control Markup Language (XACML) using blockchain technology. Using ABAC model allows the formulation of a strategy of granular access control policies. However, in the blockchain, the verbosity of XACML language

Table 2
Diagnostic evaluation to problematic approaches.

References	Approach	Implementation	Evaluation	Problem
[15]	Attribute-based encryption technique is used for implementing access control in smart grid	Yes	Computation and communication cost	Role in heterogeneous and constrained devices is ignored.
[16]	Implemented operation-based access control (an extension to the traditional role-based access control)	Yes	–	Role in heterogeneous and constrained devices is ignored.
[47]	CapBAC model for IoT environment (group access by a single token to access common services running on multiple devices)	No	–	Theoretical recommendation.
[48]	ECC-based mutual authentication and ABAC policy	Yes	Security and performance (overhead)	Complex management of access control for IoT constrained devices.
[49]	Blockchain based solution for smart home data accessing in a smart grid environment	Yes	Computation and communication cost	Role in constrained devices is ignored.
[50]	Blockchain based access control solution	Yes	Latency	Access control method strategies are missing.
[51]	Blockchain based access control solution for decentralized online social networks	Yes	Gas cost	Role in IoT environment is neglected.
[52]	Attribute-based access control and blockchain approach	Yes	Cost (gas, computation and time)	Role in IoT environment is not discussed.
[53]	Blockchain and cloud based security architecture for smart home	Yes	Computational and communication overheads	Role of constrained devices is neglected.
[54]	Blockchain and cloud based access control	Yes	Computational and communication overhead	Scalability of IoT nodes and number of requests are not discussed.
[55]	Token based access control enforced by blockchain	Yes	Computational overhead	No implementation detail of IoT devices.
[56]	Scalable solution for key and trust management of IoT devices, through blockchain and SDN	Yes	Access delay and throughput	Scaling multiple heterogeneous IoT networks
[57]	Blockchain implementation on fog nodes	Yes	Security analysis	Policies are public (Users cannot control their own privacy).
[58]	Blockchain implementation for privacy	No	–	Theoretical recommendation.
[59]	Blockchain based sub networks for software defined vehicular network	Yes	Computational cost, throughput and latency for variable number of requests	implementation detail of access control is not discussed.
[18]	SDN and blockchain based security solution	Yes	Latency and throughput	Security structure is not defined.
[60]	SDN-cloud and blockchain based verifiable authentication system	No	–	Theoretical recommendation.
[61]	Combine SDN and blockchain technology for IoT network	Yes	Scalability, defense effects, accuracy and performance (overhead)	No implementation detail about blockchain.
Proposed solution	Combine SDN and smart contract feature of blockchain technology for IoT access control	Yes	Resource access delay and throughput	–

creates a severe question of room occupancy. Authors in this research suggested a hybrid approach to tackle this problem, which stores policies directly in the blockchain but encodes this policy to a custom designed format that enables compression and avoids repetitions of information. This system, though, is limited to the usage of access management policies articulated only through the ABAC model. Since IoT systems have restricted computing resources, the scientists suggested a lightweight blockchain-based architecture [53] that nearly excluded classic blockchain's overheads while preserving most of its safety and privacy advantages. The architecture proposed used distributed trust to reduce the processing time for block validation. The proposed structure comprises of three parts: cloud storage, smart house and network overlay. As an example, a smart home scenario is used to introduce lightweight blockchain to optimize IoT security.

In [54] the researchers employed another strategy, a Blockchain-Enabled Decentralized Federated, Capacity-based Access Control (BlendCAC) system which is intended to improve IoT devices safety and privacy. In order to secure the information of smart devices, services and communication in IoT networks, it seeks to provide decentralized, scalable, fine-grained and lightweight access control alternative. A token management strategy based on identity is provided and the delegation process for federated approval is illustrated in this system design. For fully decentralized privacy management the researchers in [65,66] inaugurated a new blockchain applicability in the area of

access management by suggesting access control system, FairAccess. FairAccess allows users to control and own their data. Blockchain acts as a decentralized access control manager for this model. Unlike economic bitcoin operations, FairAccess provides creative types of operations used to request, receive, assign and erase access. Researchers set up with a Raspberry PI device and local blockchain as a proof of concept for initial implementation. For providing safe approved access to IoT assets in E2E environment, the researchers proposed IoTChain in [55], which is a combination of the Object Security Architecture for the Internet of Things (OSCAR) architecture and the Authentication and Authorization for Constrained Environments (ACE) authorization framework. IoTChain comprises of two parts, an ACE framework-based permission blockchain and an OSCAR object safety model, expanded by a collective important system. The blockchain offers a flexible and secure manner of handling requests whereas OSCAR utilizes the government ledger to manage multicast organizations for approved customers. To evaluate the feasibility and effectiveness of the proposed method, the researchers deployed the blockchain on top of personal Ethereum network.

Through blockchain-based SDN approach, in [67], ChainGuard uses SDN features to filter network traffic, thereby introducing a blockchain firewall. To determine which traffic source is valid, it communicates with the blockchain node (it guards). Packets are intercepted from illegitimate sources and therefore cannot affect the blockchain. ChainGuard considers three distinct types of remote blockchain nodes to manage remote node connections that can be valid, unlawful or unknown to the scheme (whitelist, blacklist). These three types depict the perspective of the module on the nodes with which it interacts with traffic. ChainGuard keeps track of these nodes and does not interfere with their preceding traffic forwarding. In [28], for IoT devices smart environment, the scientists implemented a blockchain-based safety layer to solve SDN safety problems. Once formed, blockchain is immutable and incorruptible. By means of blockchain, as the major distributed archive of a certain scheme, permitted nodes can immediately monitor and evaluate information produced by IoT devices once stored in that data structure, irrespective of their type or complete source amount. In [59] the researchers introduced interlinked blockchain sub networks for software defined vehicular networks to overcome the challenges of access control and authentication in a scalable environment. The researchers defined access privileges mechanism to address mobility depending upon different geographical locations. In another alternate approach the authors suggested a novel hybrid network design for the smart city in [18] by combining the features of the SDN and blockchain. The suggested architecture inherits the capabilities of both unified and hierarchical network architectures by the implementation of a hybrid technique. The researchers suggested a proof-of-work for protection of information against destruction, modification, disclosure, unauthorized access, cyberattacks and privacy in smart cities. Consequently, the novel technique of VeidBlock protocols were introduced and evaluated and proposed in [60]. All identities generated using VeidBlock strategy are verifiable and therefore confidential, in the verification and authentication process, and maintain the privacy of the user, as proof of concept, by incorporating it into an SDN cloud based infrastructure. The primary objective was to create a verifiable identity by pursuing a credible method of authentication. Entities are in control using blockchain ledger concepts, indicated as an advance mechanism for protecting from manipulation. Furthermore, in cloud environment, where all individual and interlinked devices have access to the Genesis block to create immutable blocks, and which update the currency of the blockchain, thus making Secure Gateway identification functions easy. In [24], the authors employed fog-based architecture in order to implement security services such as access control and authentication. In [23,68], the authors proposed auditable and fine-grained access control security schemes based on blockchain for Industrial IoT. In the proposed schemes, malicious attacks are prevented, and attackers can be tracked, ensuring system security. The efficiency of both research works is evaluated on the basis of overhead time and computational cost, respectively.

4. System design and architecture

Each IoT domain implementation has particular features due to which it needs its specific access control policy, which is important to be considered in the design of security solution. The main goal of our proposed framework is to address the incurring challenges of IoT associated with management, control of access to resources and security. The integration of blockchain with SD-IoT results in the resolution of policies related to intricacy, management, forgery, dissemination, access control tracking, automation, and central management, of IoT nodes. The proposed framework for implementing access control policies on IoT devices is shown in Fig. 1. On the basis of a flexibility and scalability policy-based access control solution, CapBAC (capability-based access control) strategy has been preferred as elaborated in [54]. In the capability-based access control (CapAC) model, each subject is linked with a capability list representing access permissions to all linked objects. The policies are enforced on the basis of the roles of nodes. These policies are immutable and cannot be forged or changed by the other peer devices. Access regulations and policies must be enforced by all IoT resource owners in order to prevent unauthorized use and to secure their resources and sensitive information. The proposed access control framework consists of following main components:

1. SD-IoT Controller: Interacts with the blockchain platform for management and verification, writing, tracking, storing and maintaining contracts by communicating access control policies with the smart contracts.
2. Access Control Module: Responsible for implementing SDN techniques to provide different operational and security services across the IoT network.
3. IoT Controller: Acts as a middle tier; responsible for collecting and aggregating information from IoT devices and transmitting it to application services for data analytics.
4. Blockchain platform: the permissioned Ethereum blockchain [69]; contains blocks of transactions and multiple smart contracts.
5. Sensor OpenFlow switches: Resource sufficient devices, used for implementing SDN techniques.
6. IoT smart devices registered on the same network.

The IoT controller and the SD-IoT controller both are located on the gateway which connects IoT network with the conventional internet. Smart contracts will allow the framework to implement decentralized, distributed, immutable and verifiable access control policies. The SDN controller will be responsible for providing security and privacy to the contracts. OpenFlow supported switches will be used, which separates the functionality of data plane and control plane. The controller will be responsible for high level routing decisions while data plane will be moved to the switches. Since we are using permissioned blockchain, the consensus mechanism relies on proof-of-authority (PoA). The nodes which have proven their authority can generate new blocks. To achieve this authority and a right to generate new blocks, a node must pass the network pre-defined authentication.

For this research, it is assumed that the devices are not mobile and are pre-registered on the IoT network.

As shown in Fig. 2, the access control process includes the following steps:

1. Device requests the access control module to initiate the access control policy creation process.
2. An access structure is assigned to a device by sending a transaction to deploy the contract onto the blockchain.
3. Access control policy is generated based on the access structure as defined by the access control module.
4. Requests initiated from a device are regulated in the network according to the policies.

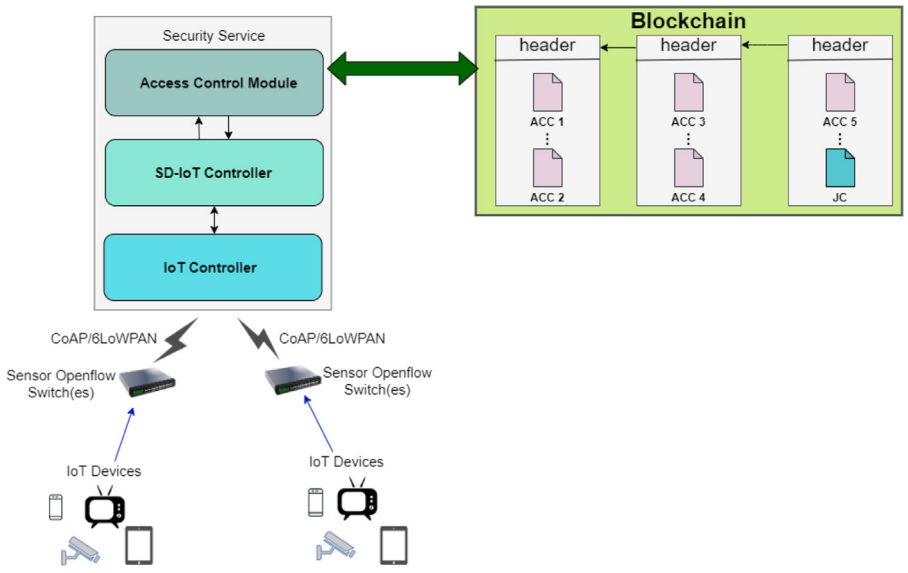

Fig. 1. Proposed smart-contract-based access control framework.

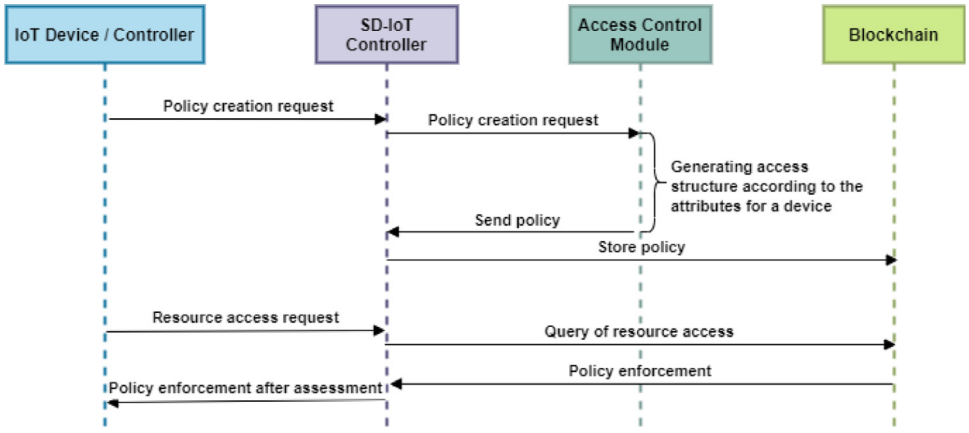

Fig. 2. Sequence diagram illustrating stepwise protocol description.

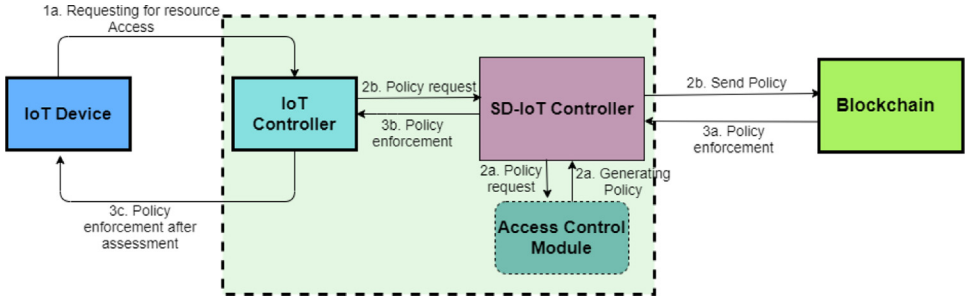

Fig. 3. Access control module.

Fig. 3 provides an overview of the access control module. The device request the IoT controller for granting resource access from a peer device. The policy request is then forwarded to SD-IoT controller, where the policy structure is defined in the access control module. The policy request is then forwarded to the blockchain from where the policy is fetched and enforced.

SD-IoT controller forwards the request to the blockchain as shown in Fig. 4, where the access control contract returns response to the register contract after verifying the subject from the judge contract. Register contract maintains the information about the judge contract and the access control contract. The register contract works as a record keeping table that maintains all the executed methods.

The proposed framework aims to resolve the intricacy of policies management, forged policies, dissemination, tracking of access control policies, automation, and central management of IoT nodes by taking into consideration the misuse and overplay, intentional or not, which impedes the efficient performance of the system. Proactive measures, but not limited to, deter such exertions include:

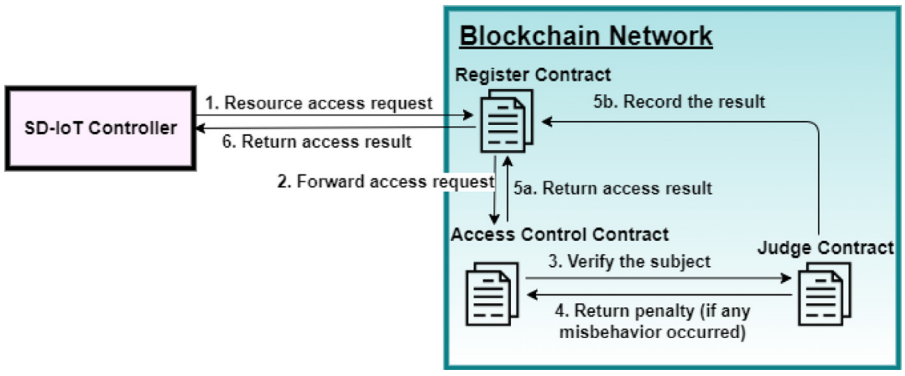

Fig. 4. Step-wise illustration of resource access request in blockchain.

Table 3
Hardware and software configuration of the system.

Item	Specification
Processor	Intel(R) Core(TM) i5 2.60 GHz
RAM	16384 MB
OS	Windows 10 Pro 64-bit
Storage	200 GB
Oracle VM VirtualBox	Version 5.2.32
VM OS	Ubuntu 18.04.2
VM RAM	8.8 GiB
VM Storage	40 GB
Truffle	5.0.14
Ganache	2.0.1

1. **Administrative role of Access Control Module**
   In case of implementation of access security, unauthorized access must be denied by the access control module in order to prevent malicious activity.
2. **Analytic role of nodes**
   The number of requests can be restricted, and penalties can be enforced according to the preferences defined by the nodes in the access control module.
3. **Scope of security**
   The level of security implementation is defined and can be enforced as per the requirement and as per the structure defined by the access control module.
4. **Traceability**
   Since the smart contracts are auditable and blockchain involves hashing function of recording the hash output of previous block, any malicious activity can be identified and is traceable. Mitigation to counter observed malicious activity to be taken up as required.
5. **Appreciating SDN approach**
   Reducing computational processing of IoT nodes related to access control of resources by the implementation of the SDN approach.

5. Prototype implementation

This section presents the detailed description of the prototype implemented for the proposed solution. We simulated our prototype environment on a laptop. The hardware and software configuration of prototype setup is listed in Table 3. The proposed solution blends SD-IoT with Ethereum based smart contract blockchain technology.

The proposed prototype consists of three types of smart contracts as can be observed in Fig. 5:

1. **Access control contracts:** Perform access control for a pair of peer nodes. The obligatory information accessed in these contracts is subject address, object address, action, permission, time of last request (ToLR). Misbehavior and the consequent penalty are also recorded, exhibited by the subject for a certain resource.
2. **Judge contract:** Receives a peer's misconduct report from an access control contract, assesses the misconduct, and determines the corresponding penalty. The basic information recorded is object address which is suffered due to occurrence of misbehavior, misbehavior, resultant penalty, and time at which misbehavior is observed.
3. **Register contract:** Stores the information about the judge contract and the access control contract. The register contract works as a lookup table that maintains all the executed methods. The information maintained in the register contract is method name, subject address, object address, corresponding smart contract name and address, creator, i.e., the peer who deployed the contract, and ABIs specified by the contract.

Each subject-object pair can have multiple access control contracts, whereas a single access contract cannot be deployed by multiple subject-object pairs. As part of the solution, a proxy server ensures communication between and integration of the blockchain and SD-IoT. The access control policy setting for the proposed prototype is illustrated in Algorithm 1. Table 4 illustrates the overall parameters used in the proposed algorithm. The object and the subject addresses are the identifiers to trigger parameters such as access resource, action and permission. The resource access request process for the proposed prototype is based on Algorithm 2, which receives access resource, action, permission and time (access request time) as input and returns the result and the penalty. From line 9 to line 13, static validation can be observed. While from line 17 up to line 22, detection of misbehavior can be noted. Dynamic array is introduced in (line 21) to store the misconducts of a particular subject. The penalty is introduced in the algorithm can be noted on line 19, where length represents the number of misbehavior of a subject, and base and interval defines the penalty change (according to the number of times penalty is enforced). The values of base and interval are initialized by judge contract when deployed. Algorithm 3 shows the method of contract registration in the Access Control Module.

Algorithm 1: Setting Access Control Policy

Input: objectAddr, subjectAddr, access_resource, p_action, permission

```
1  if Object sends policy against subject then
2      objectAddr ← getAddress(nodeObjectID)
3      subjectAddr ← getAddress(nodeObjectID)
4      sp ←
         sendPolicyParams[access_resource][p_action][permission]
5  end
```

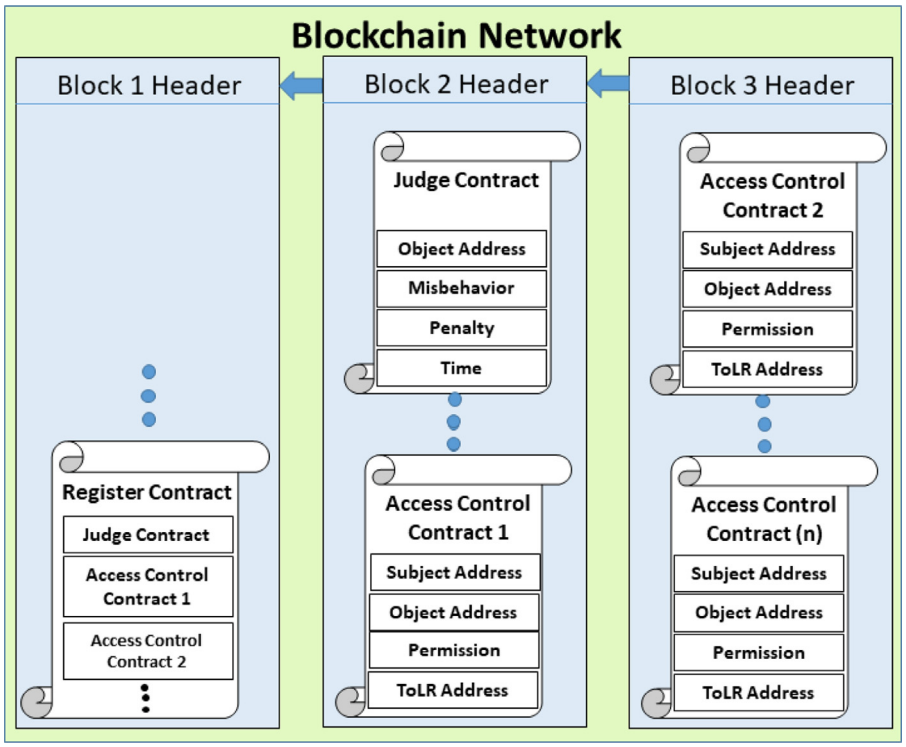

Fig. 5. Proposed smart contracts for access control.

Algorithm 2: Resource Access Request

Input: access_resource, p_action, permission, req_time
Output: result, penalty

```
if Subject sends resource access request then
    policy ← policies[access_resource][p_action]
    if timeofUnblock ≤ req_time then
        if timeofUnblock > 0 then
            policy.NoFR ← 0
            policy.ToLR ← 0
            timeofUnblock ← 0
        end
        if policy.policypermission = }}allowε then
            Policy_Check ← true
        else
            Policy_Check ← false
        end
        if req_time - policy.ToLR ≤ policy.minInterval then
            policy.NoFR ← policy.NoFR ++
            if policy.threshold ≤ policy.NoFR then
                Detecting misbehavior misb,
                behavior_check ← false,
                penalty ← base ** (length/interval)
                timeofUnblock ← p_time + penalty,
                Add misb into the resource misbehavior list,
            end
        else
            policy.NoFR ← 0
        end
    end
    policy.ToLR ← req_time
end
result ← policy_check and behavior_check
return ReturnAccessResult(result, penalty)
```

Table 4
Algorithm parameters.

Parameters	Meaning
Object	The peer node which offers resources (services).
Subject	The peer node which make resource access request.
Action	Action performed on the resource (such as read, write, execute).
Misbehavior	A simple misbehavior i.e. immensely sending resource access.
Penalty	Enforced penalty after detecting misbehavior.
NoFR	Number of frequent requests in a period of short time.
ToLR	Time of last request.
minInterval	Minimum acceptable time interval between two consecutive requests.
threshold	If NoFR is greater than or equal to the threshold, a misbehavior is identified.
timeofUnblock	Request time until it is blocked (i.e. 0 when request is unblocked).
p_time	Misbehavior detected time.
req_time	Time when request is sent.
MisbehaviorList[ ]	Misbehavior record of a particular subject on a certain resource.
methodName	A structure containing subject address, object address and resource access permissions.

In an IoT environment, devices usually have some resources (such as files, data information, storage) that other peer nodes need for performing their function properly. Access regulations and policies

Algorithm 3: Access Control Module

1 Get reference to *Register_Contract*
2 $register \leftarrow Contract(regAbi, regAddr)$
3 Use *register_contract* method to get *Access_Control* contract method
4 $methodName \leftarrow register.methods.getContract(methodName)$
5 *Add* method to the *Access_Control* Module

must be enforced by all IoT resource owners in order to prevent unauthorized use and to secure their resources and sensitive information. A device/node must have the capability to restrict access requests by unauthorized peers for querying data or storing data to prevent the illegal use of its storage space and data. The unauthorized access attempts to retrieve or monitor data can be refused by an IoT device/node.

5.1. Components

1. Smart Contract Platform: The architecture introduced is focused on the smart contract model of Ethereum. Smart contract coding is definitive and cannot be modified once it has been implemented in the blockchain. Solidity is used as language for creating smart contracts. To identify and correct as many vulnerabilities as possible, we have taken extra care in the testing process during the deployment of smart contracts.
   In order to implement the Ethereum platform in our access control framework, we need to make the following basic configurations for the system:
   - Every node must be connected to the Ethereum account, so that each node can deploy smart contract and identify itself during access control.
   - Due to the limited resources and computing power of IoT devices the Ethereum application can be run on any peer except for smart devices. All client devices are supposed to be sync on the same block.
   - Because IoT devices have no Ethereum application, by storing their accounts, IoT gates serve as agents for their local IoT devices. IoT gateways implement and manage smart contracts on behalf of IoT devices through these accounts.

   For the implementation and development of contracts Truffle (v5.0.14) [70] is used as it provides the feature of built-in-testing and a development environment for smart contracts. Truffle gives the functionality of smart contract compilation and contract deployment; and it provides an interactive console. Truffle (truffle init) sets the structure of the project. We executed truffle migrate (which automatically runs truffle compile), to deploy the contracts with the data provided in the migration files. In our prototype, to build a smart contract on the Ethereum blockchain Ganache [71], a personal blockchain platform was first installed and configured to run on default Ganache GUI IP i.e., http://127.0.0.1:7545 as a local host, as shown in Fig. 6. By default, Ganache comprises 10 accounts and 100 Ethers (crypto fuel that enables smart contracts). These accounts were used for sending and receiving Ethereum transactions as well as for smart contract operations.
2. Server: In our framework javascript based server (server.js) is used as a server proxy to help in resolving interaction and compatibility issue between the blockchain and SD-IoT. A server must be capable enough to communicate with the SD-IoT network and the blockchain in order to provide a variety of services, including sending commands and policies to executing devices for any operation, querying data or storing data in storage devices, sending resource blockchain policy and requiring node access etc.
3. SD-IoT: In our framework, for providing programmability to the IoT network, SDN-WISE controller is used. Access control module runs over this SDN-WISE controller and interacts with the blockchain to enforce immutable policies and penalty in case of any detected misbehavior.

5.2. Experimental environment

We implemented the proposed architecture between peer nodes using the above mentioned components in-order to analyze the performance of the architecture.

Conceiving an experimental setup wherein ‘node 2’ (object) is setting a simple policy against ‘node 3’ (subject), for accessing resource File A. The interval and the threshold is preset by ‘node 2’ for ‘node 3’ as depicted in Fig. 7. Violation of this policy will result into a misbehavior count.

In Fig. 8, ‘node 3’ reacts by sending a resource access request to ‘node 2’ via the controller and the blockchain. Figs. 9 and 10 illustrate the access results, wherein ‘node 3’ is permitted a misbehavior four times, whereas, a fifth misbehavior would result into a permanent blockage of access for ‘node 3’. As per policy, exceeding the threshold as instructed in Algorithm 1, blocks the resource access for that particular node.

6. Testbed and evaluation

In order to benchmark, compare and validate the performance and to observe potential bottlenecks of the proposed framework, the following two sets of experiments have been performed to investigate resource access delay and throughput:

1. Blockchain-to-Controller
   - Variable number of nodes sending requests in burst mode.
   - Variable number of nodes sending requests with 0.5 s delay.
   - Variable number of nodes sending requests with a 1 s delay.
   - Variable number of nodes sending requests with 5 s delay.
   - Fixed number of nodes (10 nodes) sending variable number of requests in burst mode.
   - Fixed number of nodes (10 nodes) sending variable number of requests with 0.5 s delay.
   - Fixed number of nodes (10 nodes) sending variable number of requests with a 1 s delay.
   - Fixed number of nodes (10 nodes) sending variable number of requests with 5 s delay.
2. End-to-End
   - Variable number of nodes sending requests in burst mode.
   - Variable number of nodes sending requests with 0.5 s delay.
   - Variable number of nodes sending requests with a 1 s delay.
   - Variable number of nodes sending requests with 5 s delay.
   - Fixed number of nodes (10 nodes) sending variable number of requests in burst mode.
   - Fixed number of nodes (10 nodes) sending variable number of requests with 0.5 s delay.
   - Fixed number of nodes (10 nodes) sending variable number of requests with a 1 s delay.
   - Fixed number of nodes (10 nodes) sending variable number of requests with 5 s delay.

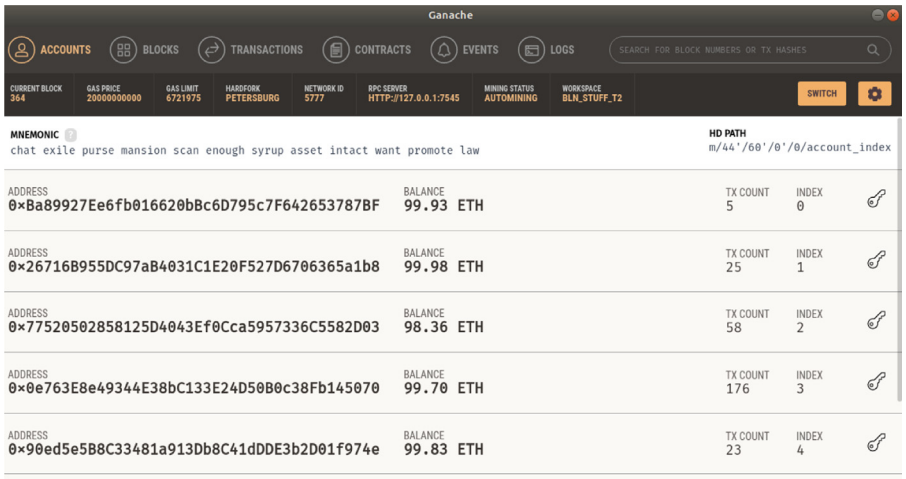

Fig. 6. Ganache platform.

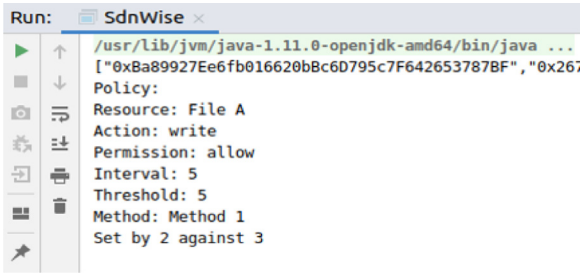

Fig. 7. Access control policy set by 'node 2' against 'node 3'.

```
Policy:
Resource: File A
Action: write
Method: Method 1
Access Request Result:
Result: true
Message: Access authorized!
Access Granted

Policy:
Resource: File A
Action: write
Method: Method 1
Access Request Result:
Result: true
Message: Access authorized!
Access Granted
```

Fig. 8. Access granted to 'node 3'.

Each task is performed three times and the average is calculated for analyzing and concluding the results. It is assumed that the devices are no mobile devices. In addition, when the network starts, all IoT devices are pre-registered on the network and no IoT device will leave the network until the network is stopped. CPU, memory and battery consumption of IoT devices is not monitored in our experiment setup.

By observing these experiments, we analyzed the bottlenecks and impact of nodes as well as the number of request in our framework. To evaluate throughput, five requests have been transmitted by each node. The *resource access delay* and the *throughput* are determined as follows:

$$\text{resource access delay} = \text{response time} - \text{request time} \tag{1}$$

$$\text{throughput} = \frac{\text{number of requests}}{\text{last request time} - \text{first request time}} \tag{2}$$

```
Policy:
Resource: File A
Action: write
Method: Method 1
Access Request Result:
Result: false
Message: Misbehavior detected!
Penalty: 2 minutes!
Access Not Granted

Policy:
Resource: File A
Action: write
Method: Method 1
Access Request Result:
Result: false
Message: Requests are blocked!
Access Not Granted
```

Fig. 9. Misbehavior detected and penalty is executed against 'node 3'.

```
Policy:
Resource: File A
Action: write
Method: Method 1
Access Request Result:
Result: false
Message: Misbehavior detected!
Penalty: 8 minutes!
Access Not Granted

Policy:
Resource: File A
Action: write
Method: Method 1
Access Request Result:
Result: false
Message: Requests are blocked!
Access Not Granted
```

Fig. 10. Penalty after fourth time and fifth time for 'node 3'.

6.1. Blockchain-to-controller

In these experiments, nodes request for getting access to a resource from a peer node. The throughput and resource access delay between the blockchain and the controller are observed and analyzed.

6.1.1. Variable number of nodes sending requests in burst mode

A variable number of nodes were made to send access requests to the blockchain in a burst manner, keeping constant the number of

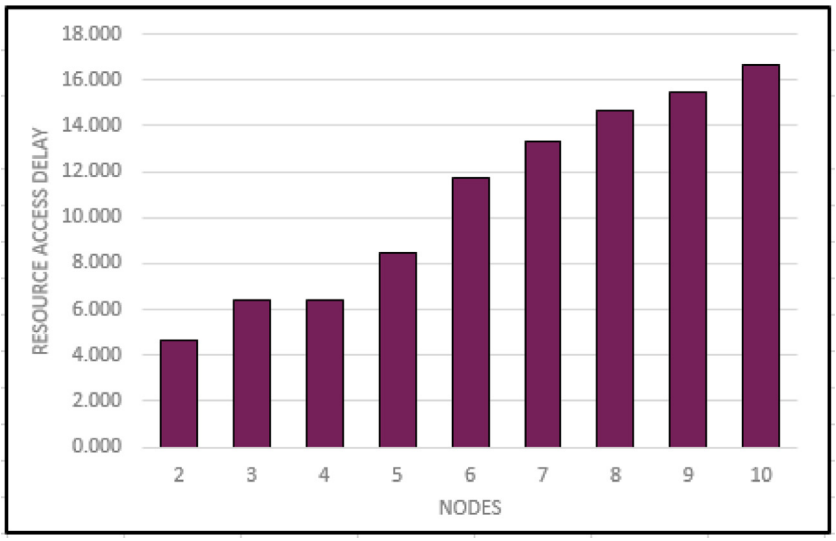

Fig. 11. Resource access delay for different number of nodes in burst mode, observed between the blockchain and the controller.

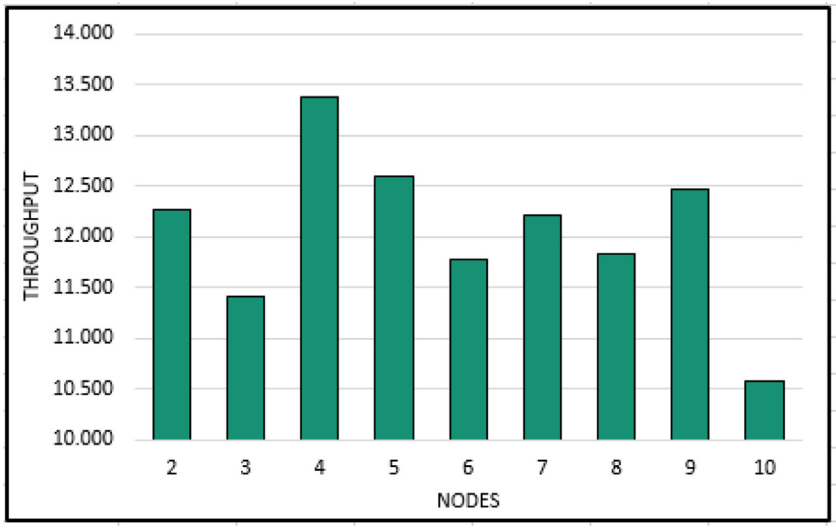

Fig. 12. Throughput for different number of nodes in burst mode, observed between the blockchain and the controller.

Table 5
Throughput and resource access delay in burst mode for variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	4.668	12.272
3	6.379	11.416
4	6.383	13.372
5	8.437	12.584
6	11.770	11.776
7	13.362	12.216
8	14.687	11.837
9	15.489	12.459
10	16.679	10.568

request per each node, i.e., five requests per node. Analyzing the results of Table 5, it is evident that the impact of throughput is not regular in this scenario, whereas a gradual increase in resource access delay is observed.

In Fig. 11, the graph depicts number of nodes on the x-axis and corresponding resource access delays on the y-axis. If the number of requests is kept constant, i.e., five per node, there is a proportional increase in resource access delay with an increase in the number of nodes. Fig. 12 represents the number of nodes versus the corresponding throughputs. With the number of requests kept constant to five for each node, in burst mode, with the increase in number of nodes, inconsistent variation of throughput is observed.

6.1.2. Variable number of nodes sending requests with 0.5 s delay

A variable number of nodes were made to transmit access requests with a 0.5 s delay to the blockchain, keeping constant the number of requests for each node, i.e., five requests per node. Analyzing the results of Table 6, it is evident that with the increase in number of nodes, there is a steady increase in resource access delay, whereas only a negligible change in throughput can be observed.

Fig. 13, depicts number of nodes versus resource access delays, keeping the number of requests to five per node. With an increase in the number of nodes an orderly incremental change in resource access delay is evident. Fig. 14, depicts the number of nodes versus throughputs, with an enforcement of 0.5 s network delay and five requests per node, the inconsistent variation in throughput to the regular increase in nodes seems obvious.

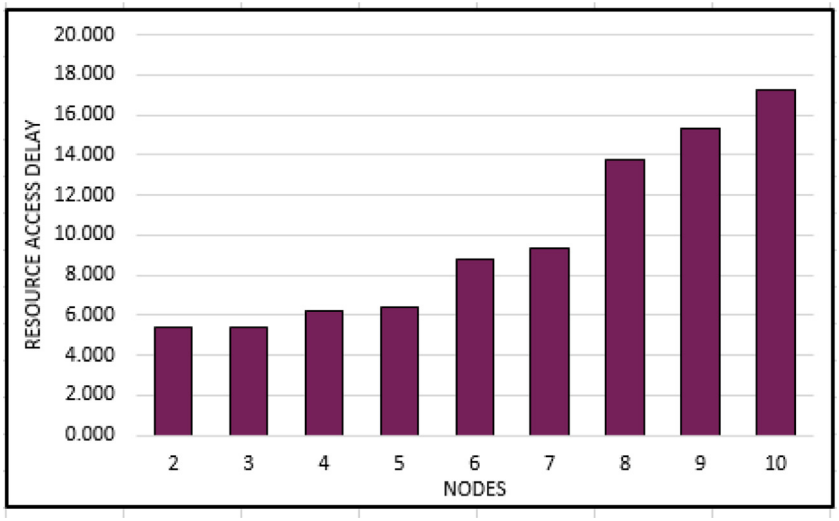

Fig. 13. Resource access delay for different number of nodes when delay is 0.5 s, observed between blockchain to controller.

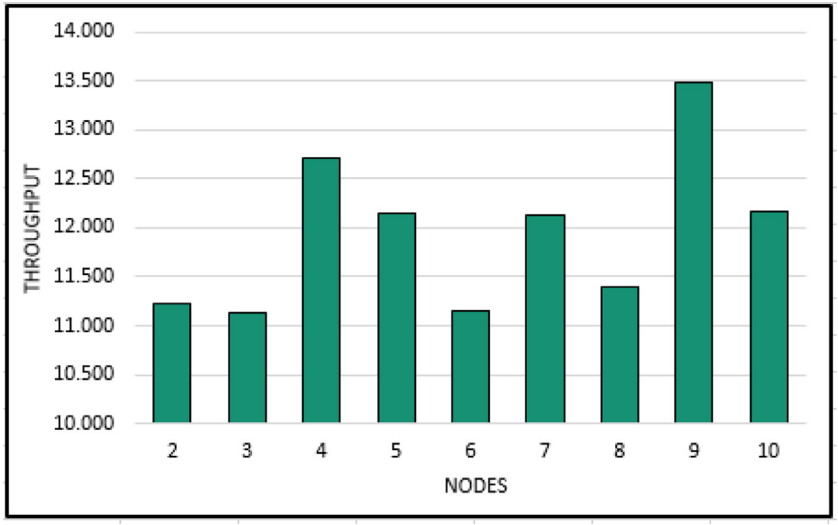

Fig. 14. Throughput for different numbers of nodes with a delay of 0.5 s, observed between the blockchain and the controller.

Table 6
Throughput and resource access delay with 0.5 s delay for a variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	5.363	11.227
3	5.379	11.126
4	6.226	12.712
5	6.397	12.154
6	8.770	11.148
7	9.362	12.122
8	13.787	11.387
9	15.329	13.486
10	17.279	12.158

Table 7
Throughput and resource access delay with 1 s delay for variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	6.241	12.272
3	6.574	12.416
4	8.143	12.372
5	10.594	12.442
6	14.787	11.776
7	16.891	12.216
8	16.961	13.624
9	28.342	12.956
10	32.782	14.165

6.1.3. Variable number of nodes sending requests with 1 s delay

In the control setting to the blockchain, having five requests per node and delay of 1 s in-between requests, the resulting values obtained, as shown Table 7, indicate that the impact of throughput is random and does not depends upon the number of nodes. Whereas the change observed in resource access delay is multiple with each change in nodes.

This is evident in the graphic representation of Fig. 15. With the x and y axis representing the nodes and resource access delay respectively. The increase in resource access delay with the respective increase in nodes can be easily perceived. In a similar controlled environment, having network delay of 1 s and authorizing five requests per node, and inconsistent change in throughput, with the increase of nodes, is observed and depicted in Fig. 16.

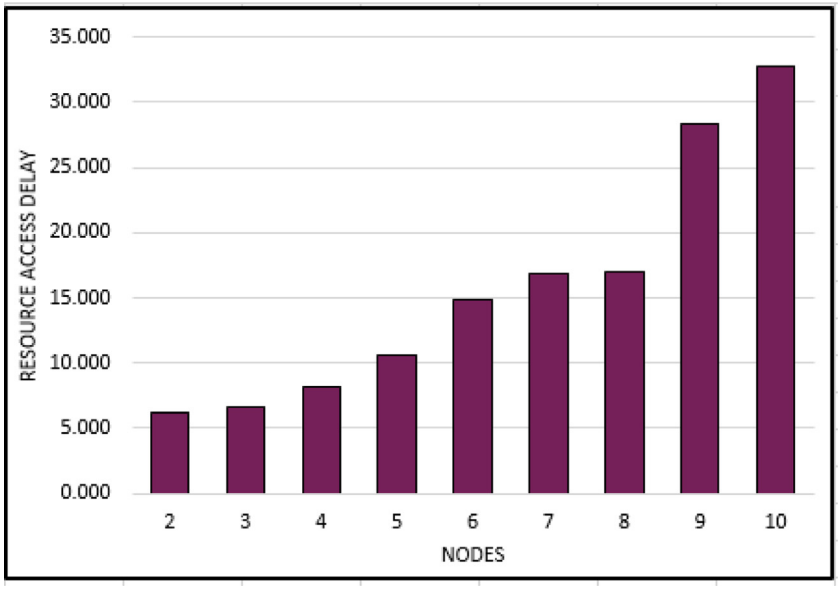

Fig. 15. Resource access delay for each node when delay is 1 s, observed between blockchain to controller.

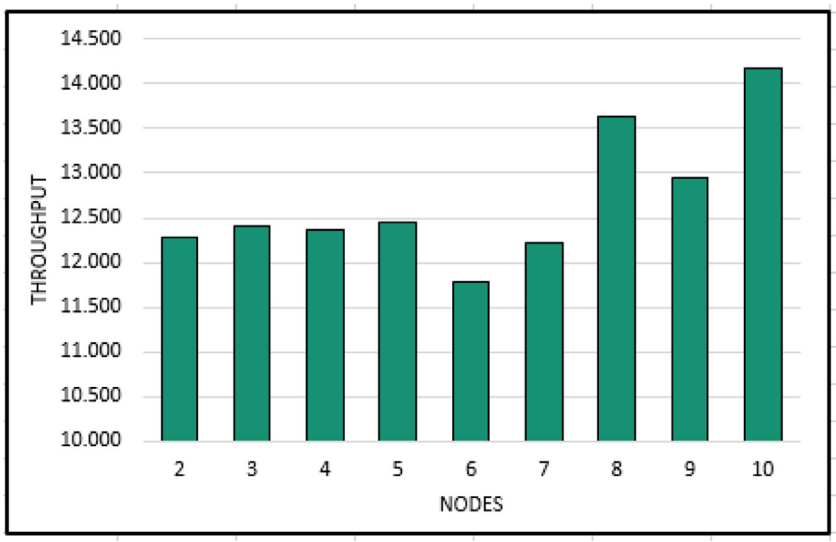

Fig. 16. Throughput for each node when delay is 1 s, observed between blockchain to controller.

6.1.4. Variable number of nodes sending requests with 5 s delay

With the nodes limited to five requests each and a network delay of 5 s, the blockchain is influenced by access requests. Table 8 clearly depicts an increase in resource access delay effected with the increase in number of nodes. Comparatively the change in throughput is hardly significant.

The same is represented as in Fig. 17, having nodes on x-axis and resource access delay on the y-axis, with similar control of five requests per node and network delay of 5 s, the graphic results display an increase resource access delay with the systematic increase in number of nodes. However, significant variation of throughput value against increase of nodes is shown in Fig. 18. The graph represents nodes on the x-axis and throughput on the y-axis, and number of requests limited to five for each node and 5 s delay in between.

Table 8
Throughput and resource access delay with 5 s delay for variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	12.241	13.587
3	18.574	11.608
4	23.143	13.919
5	24.594	13.845
6	34.787	12.827
7	36.891	12.797
8	37.461	13.693
9	48.342	12.441
10	56.782	12.704

6.1.5. Fixed number of nodes sending a variable number of requests in burst mode

The tabular information in Table 9 projects the impact of throughput as independent of number of requests per node, and similarly a minimal change in resource access delay is noted. Herein the number of nodes was kept constant as 10, with the systematic increase of 5 in number of requests ranging from 10 to 45 per node, delivered in burst mode.

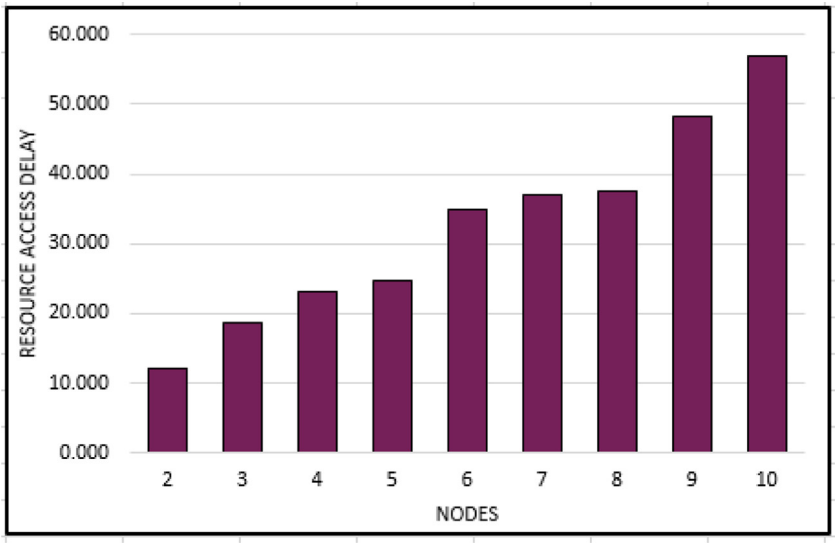

Fig. 17. Resource access delay for each node when delay is 5 s, observed between blockchain to controller.

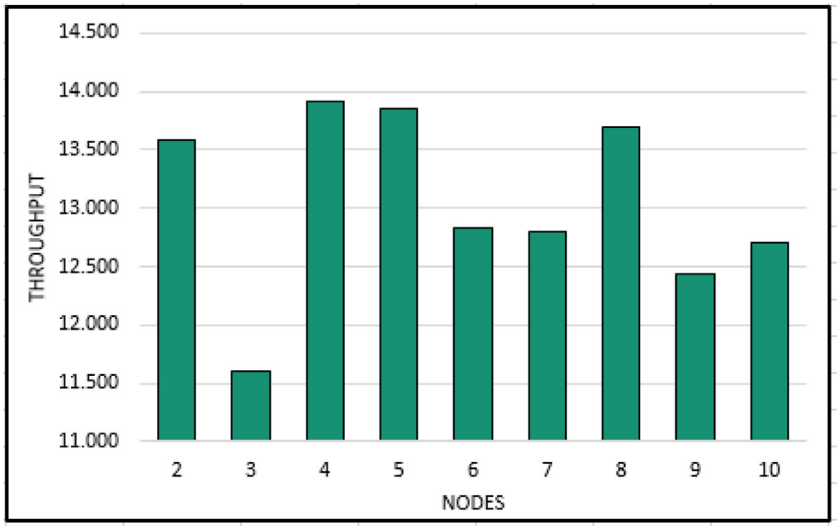

Fig. 18. Throughput for each node when delay is 5 s, observed between blockchain to controller.

Table 9
Throughput and resource access delay in burst mode for 10 nodes.

Requests	Resource access delay	Throughput (no. of req/time)
10	15.092	1.294
15	14.492	1.249
20	15.196	1.226
25	15.165	1.199
30	15.094	1.195
35	14.978	1.191
40	15.121	1.204
45	15.236	1.214

Table 10
Throughput and resource access delay for 10 nodes with 0.5 s delay.

Requests	Resource access delay	Throughput (no. of req/time)
10	17.122	1.244
15	17.412	1.149
20	16.146	1.246
25	16.165	1.225
30	15.894	1.265
35	16.248	1.142
40	16.221	1.104
45	16.763	1.215

The graph as in Fig. 19, depicts number of requests on the *x*-axis and resource access delay on the *y*-axis. It is observed that on resource access delay the impact of the number of request per node is small. In Fig. 20, a similar comparison between requests, on *x*-axis, and throughput presented on *y*-axis shows a parabolic reduction in throughput, and a gradual increase after the 35 request mark.

6.1.6. Fixed number of nodes (10 nodes) sending a variable number of requests with 0.5 s delay

In this scenario, with a 5 number of gradual increase and a 0.5 s delay in between each request to a fixed set of 10 nodes, Table 10 displays a steady decrease in resource access delay before initiating an upward trend halfway through. The variation in throughput observation appears insignificant.

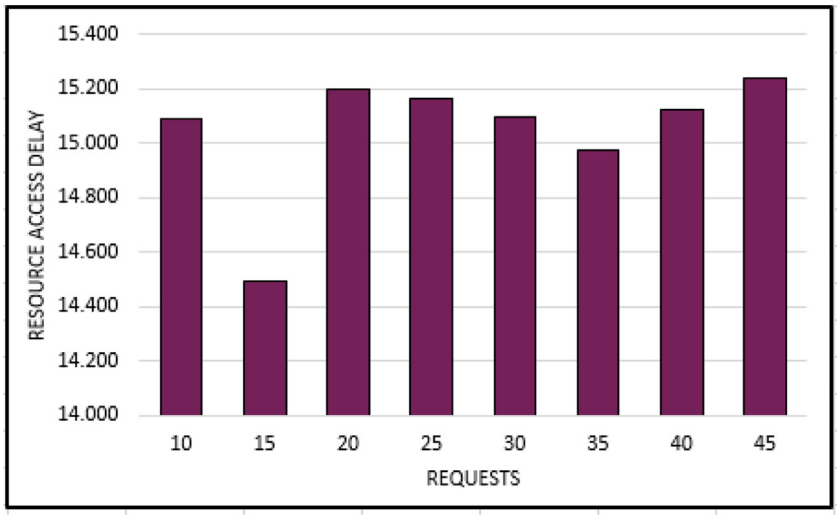

Fig. 19. Variations of resource access delay for 10 node sending requests in burst mode, observed between blockchain to controller.

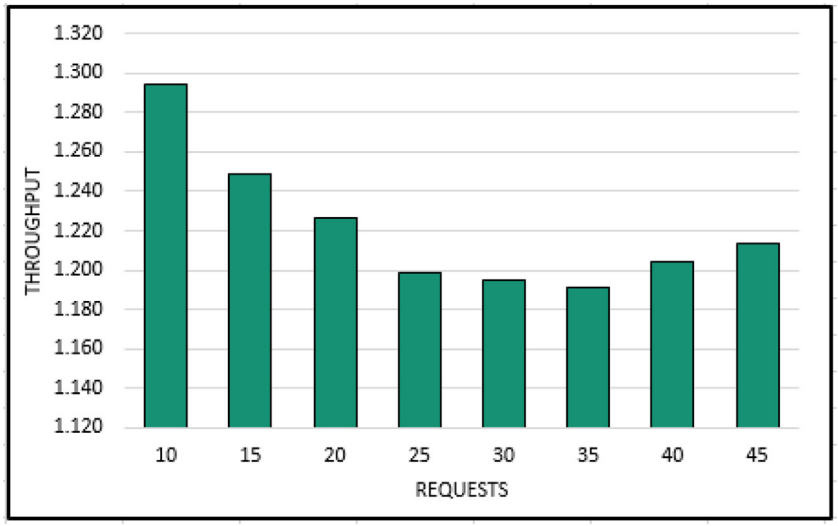

Fig. 20. Variations of throughput for 10 node sending requests in burst mode, observed between blockchain to controller.

The graphic representation of the above table (Table 10) in Fig. 21 presents the requests transmitted on x-axis and resource access delay on y-axis. The initial steady decrease in resource access delay until halfway through is observable. In Fig. 22, the graph supports the observations between requests on x-axis, and throughput on y-axis clarifying initial observations of a disproportionate and inconsistent change in relation to each other when the number of nodes is fixed as 10.

6.1.7. Fixed number of nodes (10 nodes) sending a variable number of requests with 1 s delay

Table 11 displays the results of an environment wherein 1 s network delay is imposed to a fixed number of 10 nodes, and a gradual increase of 5 requests per node is initiated to the blockchain. Analysis of the results indicates random variation of resource access delay whereas the resultant throughput is negligible.

With Fig. 23 presenting number of requests per node on the x-axis and resource access delay on the y-axis, the resultant graph projects the same observations as envisaged from the Table 11 above, and the disorderly variation is evident. With the number of request and throughput placed on x and y axis respectively, as shown in Fig. 24, the initial observations of inconsistent and negligible results is obvious.

Table 11
Throughput and resource access delay for 10 nodes with 1 s delay.

Requests	Resource access delay	Throughput (no. of req/time)
10	24.138	1.288
15	23.795	1.152
20	24.131	1.155
25	24.161	1.249
30	23.524	1.247
35	24.163	1.165
40	25.121	1.651
45	25.454	1.243

6.1.8. Fixed number of nodes sending a variable number of requests with 5 s delay

Table 12 presents a setup where 5 s network delay is imposed on a constant number of nodes i.e. 10, ensuring variable request transmission to the blockchain. Analyzing the results it is observed that the gains in the throughput, although minimal, are gradual initially with rapid increase of request exceeds 35. However the comparative data of resource access delay depicts negligible impact when the number of nodes is less than 20 requests mark, subsequently showing a substantial jump and to gradual increase thereafter.

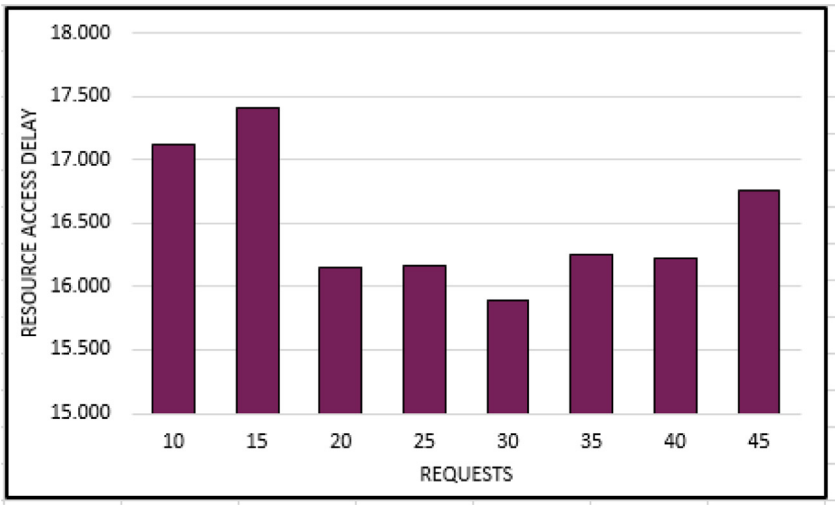

Fig. 21. Resource access delay for 10 nodes when delay is 0.5 s, observed between blockchain to controller.

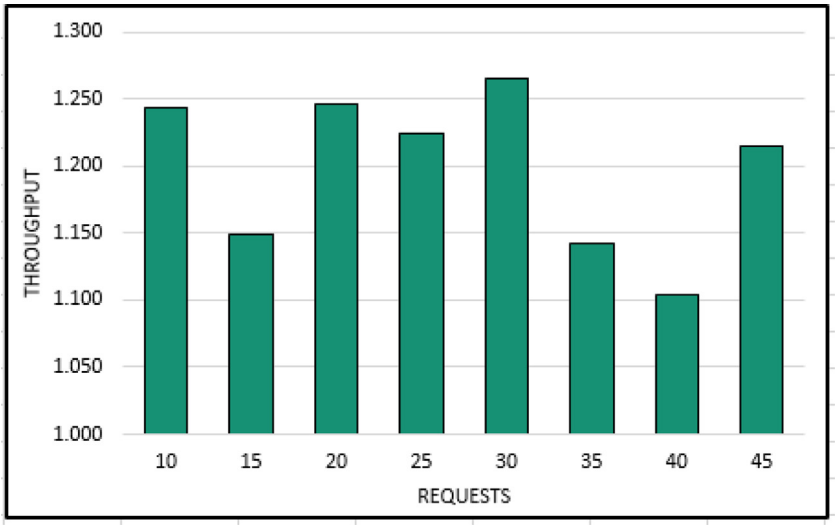

Fig. 22. Throughput for 10 nodes when delay is 0.5 s, observed between blockchain to controller.

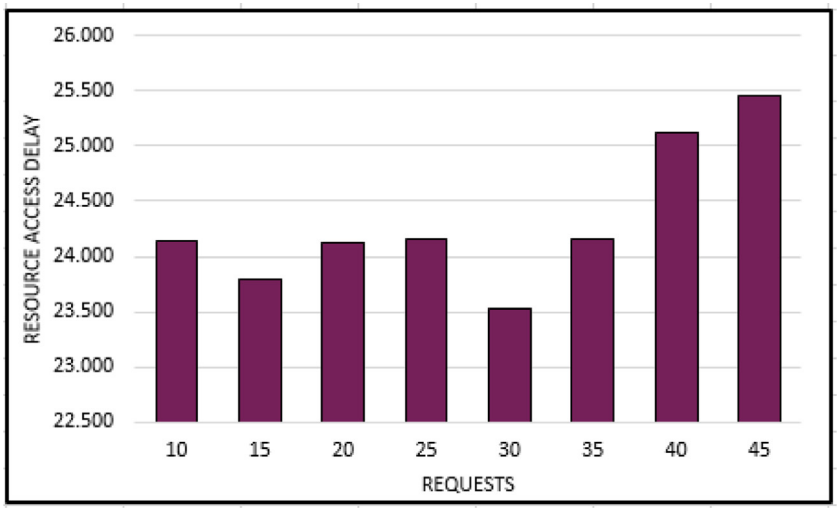

Fig. 23. Variation of resource access delay for 10 nodes when delay is 1 s, observed between blockchain to controller.

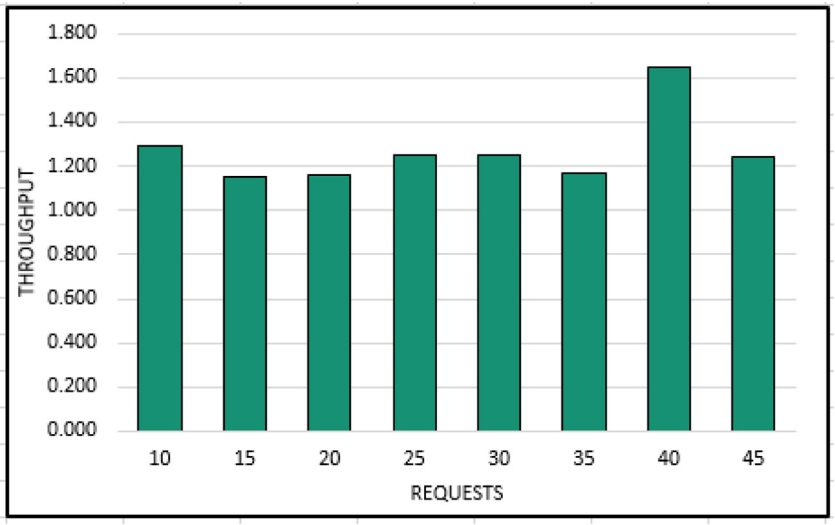

Fig. 24. Throughput for 10 nodes when delay is 1 s, observed between blockchain to controller.

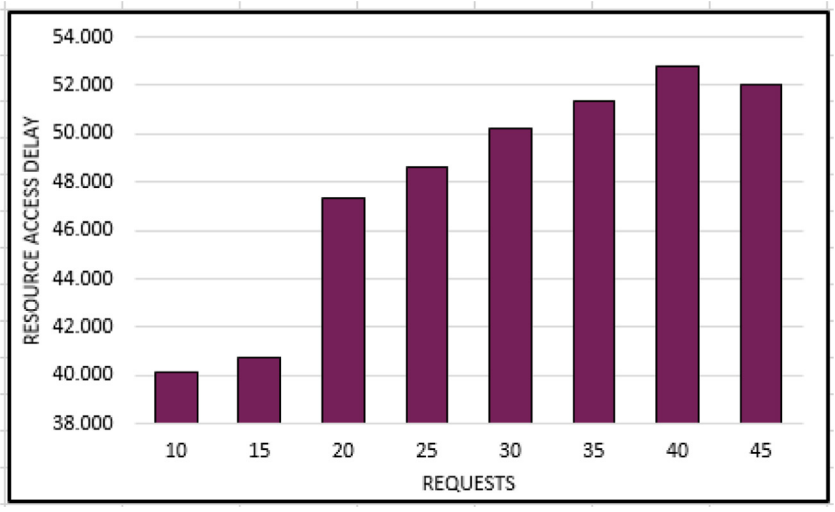

Fig. 25. Resource access delay for 10 nodes when delay is 5 s, observed between blockchain to controller.

Table 12
Throughput and resource access delay for 10 nodes with 5 s delay.

Requests	Resource access delay	Throughput (no. of req/time)
10	40.178	1.188
15	40.725	1.191
20	47.312	1.195
25	48.609	1.199
30	50.241	1.205
35	51.335	1.214
40	52.817	1.251
45	52.052	1.298

Table 13
Throughput and resource access delay in burst mode for variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	8.729	13.289
3	8.827	13.876
4	12.906	13.479
5	18.990	13.584
6	24.020	13.258
7	25.705	14.213
8	32.890	13.458
9	36.684	12.958
10	38.320	13.238

The tabular observations can be clearly identified in the graphic representations, with the requests on x-axis and resource access delay on y-axis in Fig. 25. The earlier observations can be identified easily. In Fig. 26, a similar graphical representation of Table 12 signifies the rapid increase in throughput after the number of request increase 35.

6.2. End-to-end

In this set of experiments case, the resource accessing request is sent from node to node (end-to-end), for observing and analyzing the comparative end to end performance benchmarks (throughput and resource access delay).

6.2.1. Variable number of nodes sending requests in burst mode

Application of burst mode, but limiting five request per node, Table 13 give the analysis that increasing the number of nodes also facilitates continuous increase in resource access delay but the throughput variation is negligible, in an end to end application.

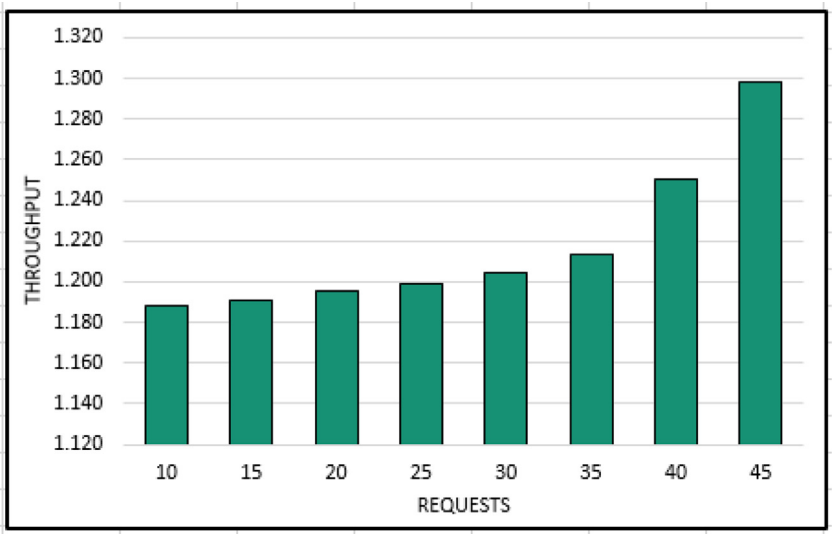

Fig. 26. Throughput for 10 nodes when delay is 5 s, observed between blockchain to controller.

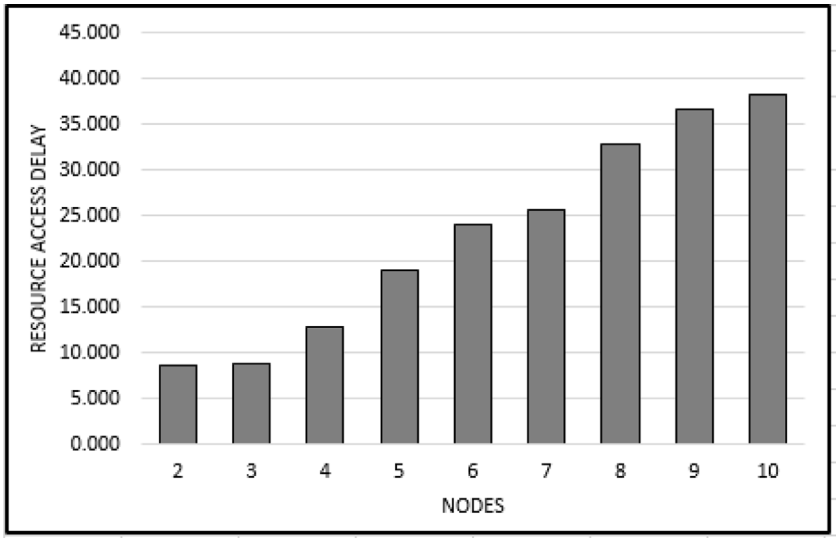

Fig. 27. Variations of resource access delay for each node in burst mode, observed between node to node.

The graphical representation of the above Table 13 is clearly substantiated in Figs. 27 and 28, wherein the steady increase in nodes is depicted on x-axis and the y-axis indicate resource access delay and throughput respectively. As can be observed that with the increase in the nodes, there is a significant increase in resource access delay and the throughput varies for each node in burst mode.

6.2.2. Variable number of nodes sending requests with 0.5 s delay

Inducing 0.5 s delay and keeping the rate constant at five request per node, Table 14 signifies the increase in nodes shows substantial increase in resource access delay, however the throughput results indicated as irregular. The analysis is distinctively visible when presented in graphically in Figs. 29 and 30.

Table 14
Throughput and resource access delay with 0.5 s delay for variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	7.426	12.219
3	8.217	11.166
4	9.906	12.439
5	10.160	13.521
6	16.150	13.522
7	20.241	12.917
8	24.120	13.212
9	32.121	12.128
10	36.310	13.136

6.2.3. Variable number of nodes sending requests with 1 s delay

Inducing 1 s delay and keeping the rate of request constant at five, in an end to end scenario, the analysis presented in Table 15 signifies a rhythmic increase in resource access delay but an irregular throughput observation. The same can be explicitly observed in the graphical representation of each as in Figs. 31 and 32. It can be observed that when network delay of 1 s is imposed the impact of throughput does not depend on the number of nodes. Although, by increasing the number of nodes, a significant increase in resource access delay has been noted.

6.2.4. Variable number of nodes with 5 s delay

In an end to end scenario, inducing 5 s delay and maintaining the request constant, the analysis presented in Table 16 depicts significant increase in resource access delay while irregularity is observed

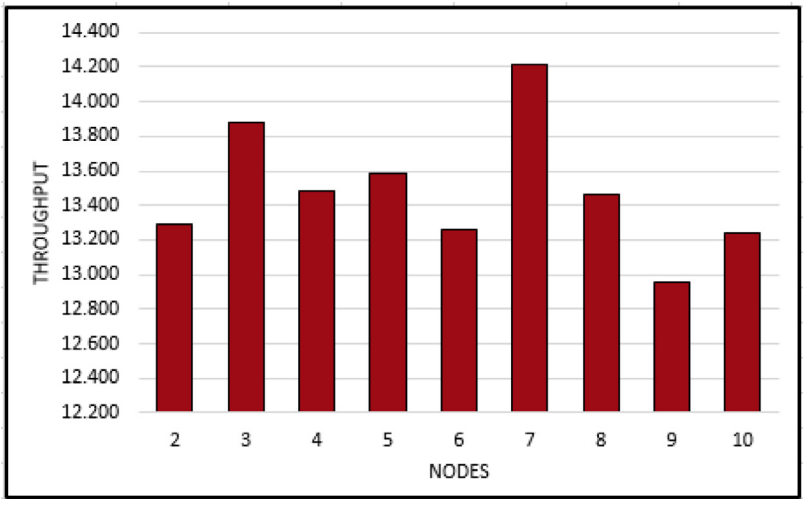

Fig. 28. Throughput for each node sending request in burst mode, observed between node to node.

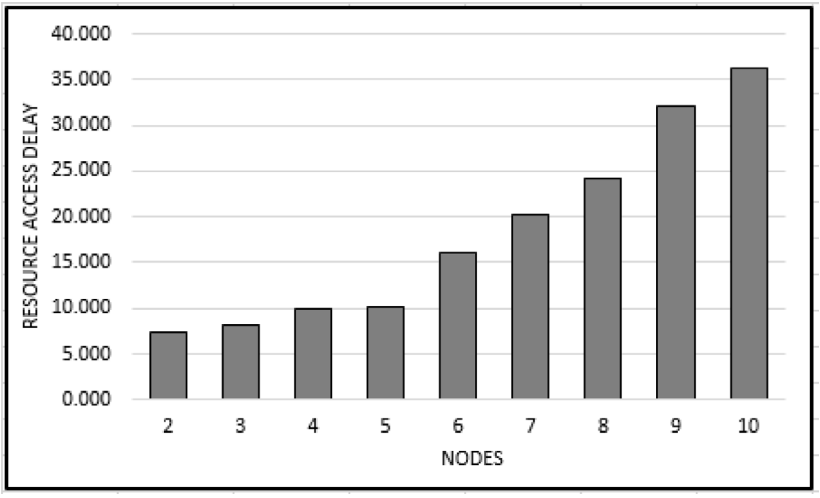

Fig. 29. Variation of resource access delay for each node when delay is 0.5 s, observed between node to node.

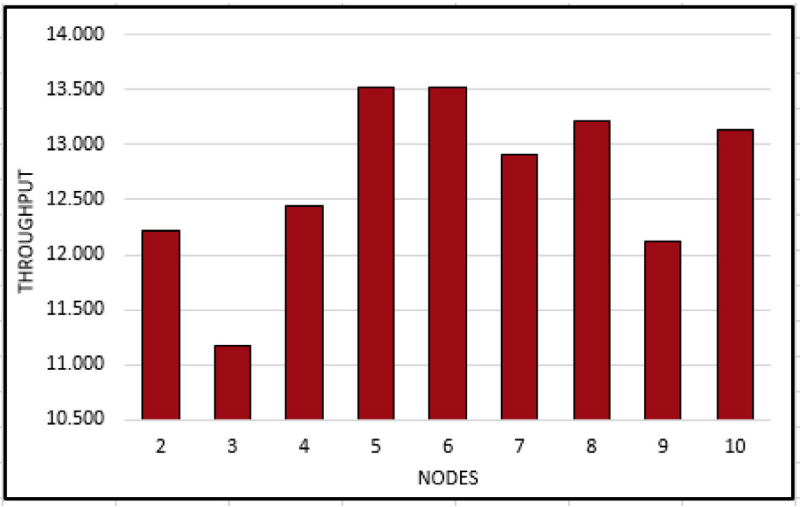

Fig. 30. Variation of throughput for each node when delay is 0.5 s, observed between node to node.

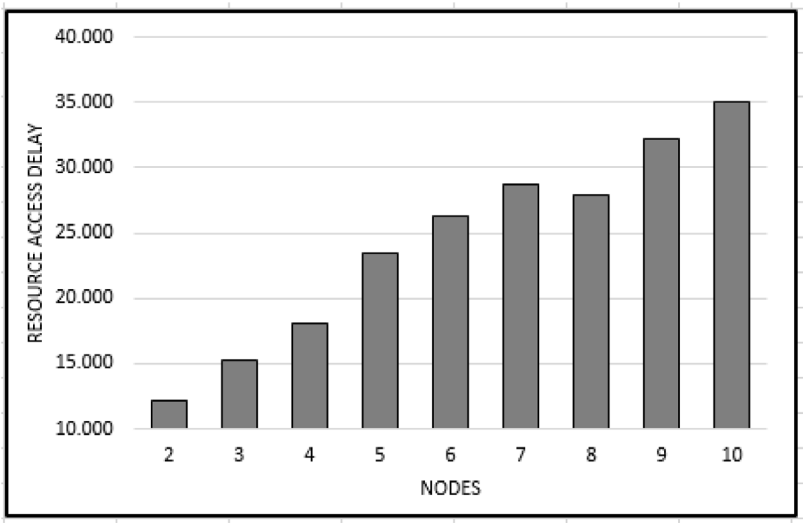

Fig. 31. Resource access delay for each node when delay is 1 s, observed between node to node.

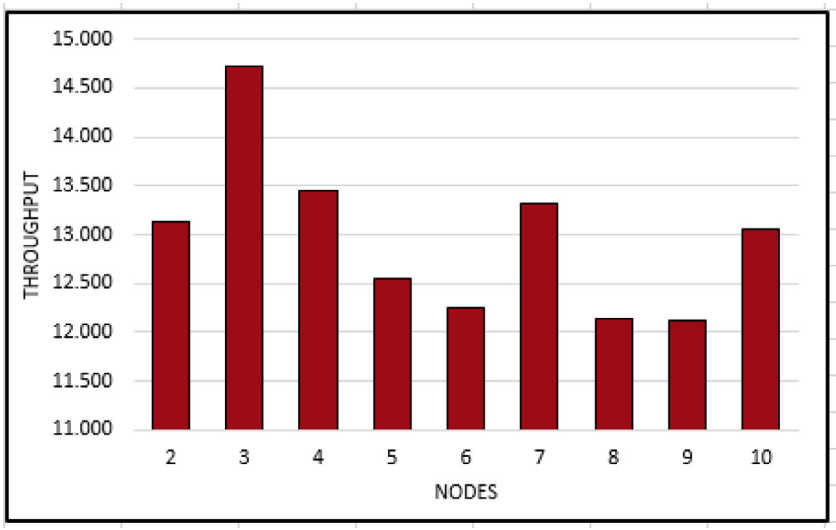

Fig. 32. Throughput for each node when delay is 1 s, observed between node to node.

Table 15
Throughput and resource access delay with 1 s delay for variable number of nodes.

Nodes	Resource access delay	Throughput (no. of req/time)
2	12.216	13.128
3	15.315	14.716
4	18.123	13.443
5	23.459	12.548
6	26.275	12.246
7	28.766	13.312
8	27.908	12.148
9	32.275	12.118
10	35.103	13.058

Table 16
Throughput and resource access delay for variable number of nodes with 5 s delay.

Nodes	Resource access delay	Throughput (no. of req/time)
2	16.666	13.289
3	18.385	13.876
4	27.654	13.479
5	29.459	12.783
6	30.275	14.252
7	38.766	14.213
8	44.909	13.548
9	46.992	13.582
10	56.048	13.823

in throughput. Figs. 33 and 34 graphically represents the perceived analysis.

6.2.5. Fixed number of nodes sending a variable number of requests in burst mode

In this scenario, the number of nodes i.e. 10 remained constant while the number of access requests varies and transmitted to the peer node in a burst manner. By analyzing the results of Table 17, the stable tendency of resource access delay across variable number of requests is observable, whereas for throughput the smooth variations appear systematic at every observation. Graphical representation of this scenario can be observed in Figs. 35 and 36 which validate the tabular observations.

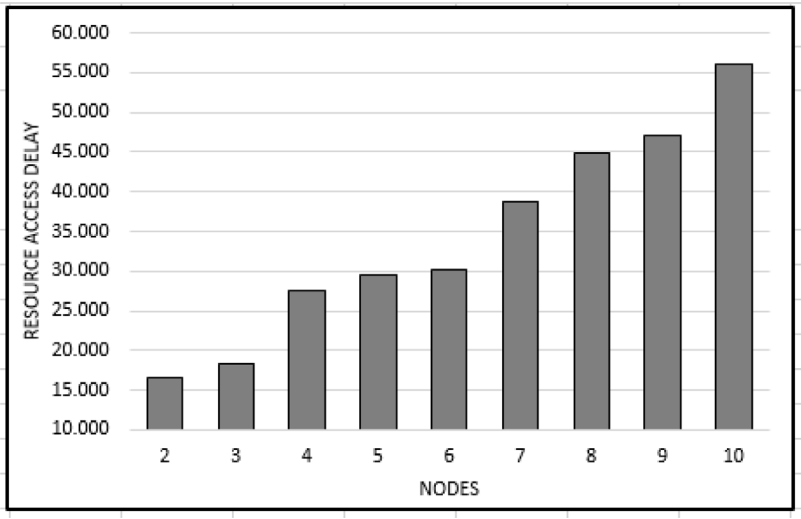

Fig. 33. Resource access delay for each node when delay is 5 s, observed between node to node.

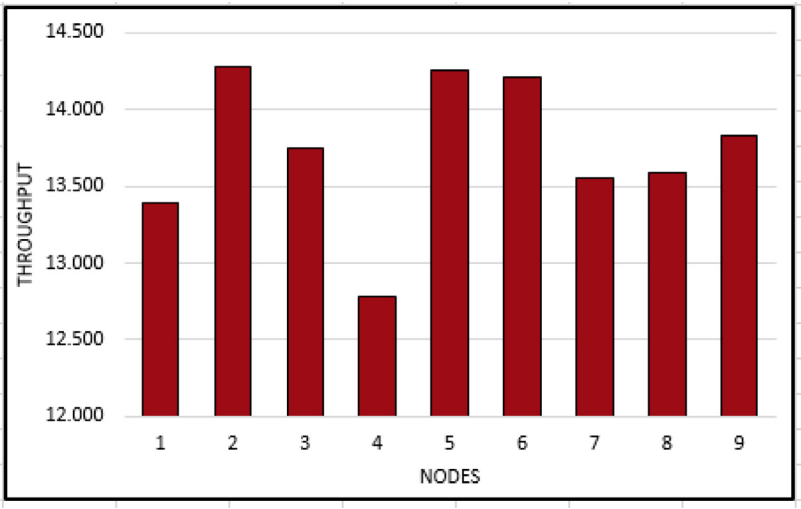

Fig. 34. Throughput for each node when delay is 5 s, observed between node to node.

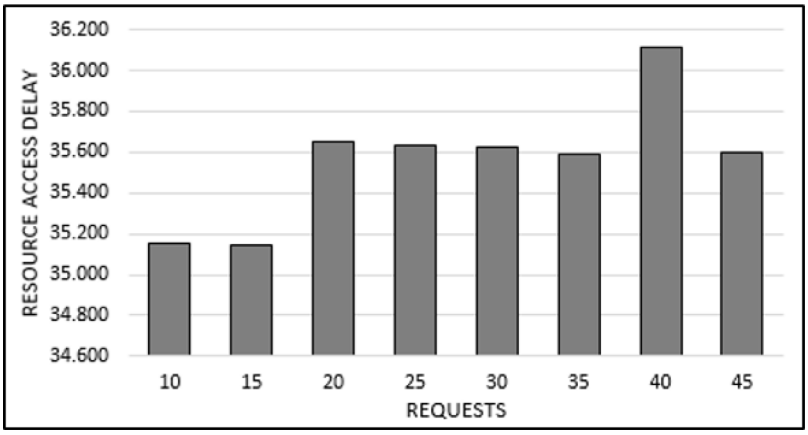

Fig. 35. Variations of resource access delay for 10 node sending requests in bursted mode, observed between node to node.

6.2.6. Fixed number of nodes sending the variable number of requests with 0.5 s delay

Table 18 presents an environment wherein a 0.5 s network delay when imposed on 10 constant nodes in an end to end environment while the number of access requests varies when transmitted to the blockchain. By analyzing the results it can be observed that the impact of throughput and resource access delay is small when the number of requests per node varies respectively. The same is better analyzed

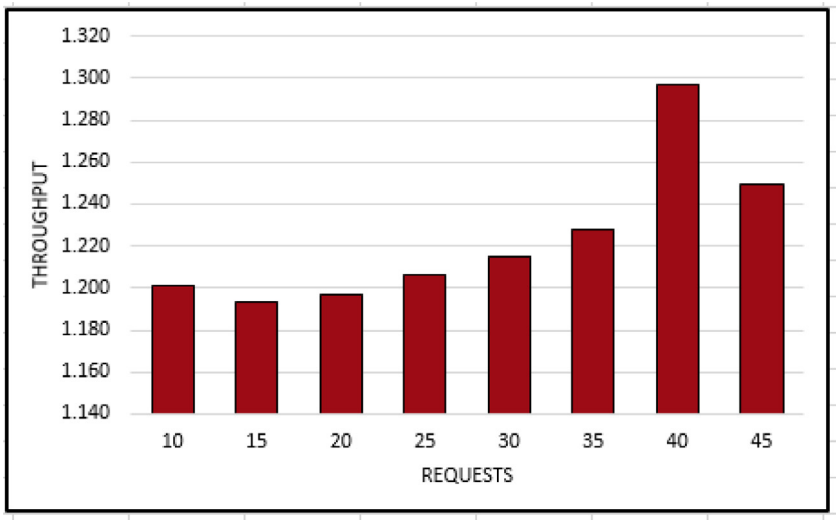

Fig. 36. Variations of throughput for 10 node sending requests in bursted mode, observed between node to node.

Table 17
Throughput and resource access delay in burst mode for 10 nodes.

Requests	Resource access delay	Throughput (no. of req/time)
10	35.155	1.201
15	35.14	1.194
20	35.65	1.197
25	35.637	1.207
30	35.624	1.215
35	35.586	1.228
40	36.118	1.297
45	35.598	1.249

Table 18
Throughput and resource access delay for 10 nodes with 0.5 s delay.

Requests	Resource access delay	Throughput (no. of req/time)
10	37.886	1.251
15	38.714	1.939
20	38.165	1.270
25	38.186	1.247
30	37.624	1.225
35	37.465	1.128
40	37.452	1.267
45	38.197	1.244

Table 19
Throughput and resource access delay for 10 nodes with 1 s delay.

Requests	Resource access delay	Throughput (no. of req/time)
10	40.124	1.231
15	40.725	1.245
20	41.721	1.163
25	41.609	1.275
30	44.124	1.127
35	41.335	1.205
40	43.177	1.262
45	44.725	1.246

when visualized as in Figs. 37 and 38. The resultant observation to resource access delay were randomized and inconsistent. However the throughput behavior was almost negligible in change except an undetermined surge at 15 request mark.

Table 20
Throughput and resource access delay for 10 nodes with 5 s delay.

Requests	Resource access delay	Throughput (no. of req/time)
10	46.589	1.191
15	46.302	1.195
20	49.147	1.199
25	49.321	1.205
30	50.308	1.213
35	50.953	1.225
40	51.695	1.252
45	51.282	1.296

6.2.7. Fixed number of nodes sending the variable number of requests with 1 s delay

When a 1 s network delay is imposed and the number of nodes is kept constant to 10, and the transmission of request to the end node varied, the exertion presents an irregular increase of resource access delay whereas the throughput results are indistinct as shown in Table 19. This scenario is better understandable when presented graphically as in Figs. 39 and 40.

6.2.8. Fixed number of nodes sending the variable number of requests with 5 s delay

Table 20 presents an almost gradual resulting effect to throughput against resource access delay in an environment wherein a 5 s network delay is imposed and the number of nodes kept constant at 10, while transmitting variable requests to end peer node. Graphically presenting the request on x-axis and alternating resource access delay and throughput, as in Figs. 41 and 42, verifies the numerical observations.

6.3. Analysis of results and comparisons

The comparative evaluations on the above data, consisting resource access delay and throughput, both for variable and fixed number of nodes, utilizing multitude probable environments created in blockchain-to controller and end-to-end scenarios is analyzed below.

6.3.1. Blockchain-to-controller

As observed in Figs. 43 and 44, comparing resource access delay and throughput for variable number of nodes in burst mode and imposing 0.5, 1, 5 s delay respectively, show the rate at which these operations are carried out. Fig. 43 depicts that increase in nodes significantly impacts in increase in resource access delay while increasing the nodes

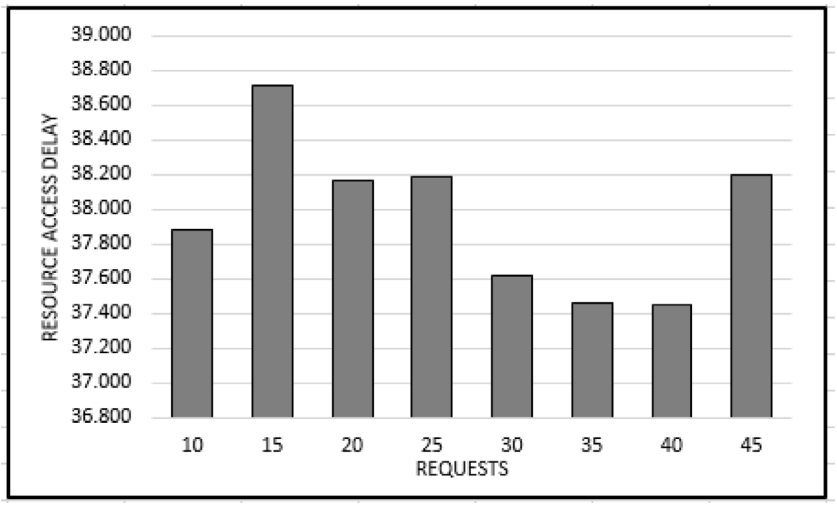

Fig. 37. Resource access delay for 10 nodes when delay is 0.5 s, observed between node to node.

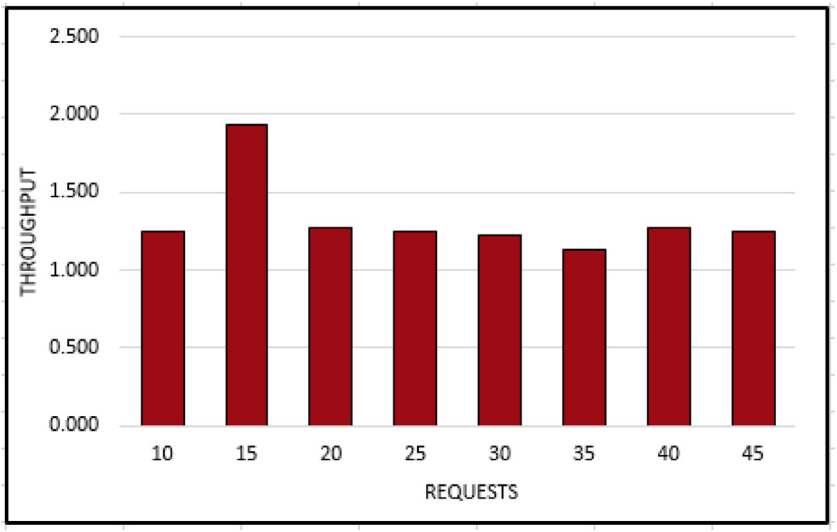

Fig. 38. Throughput for 10 nodes when delay is 0.5 s, observed between node to node.

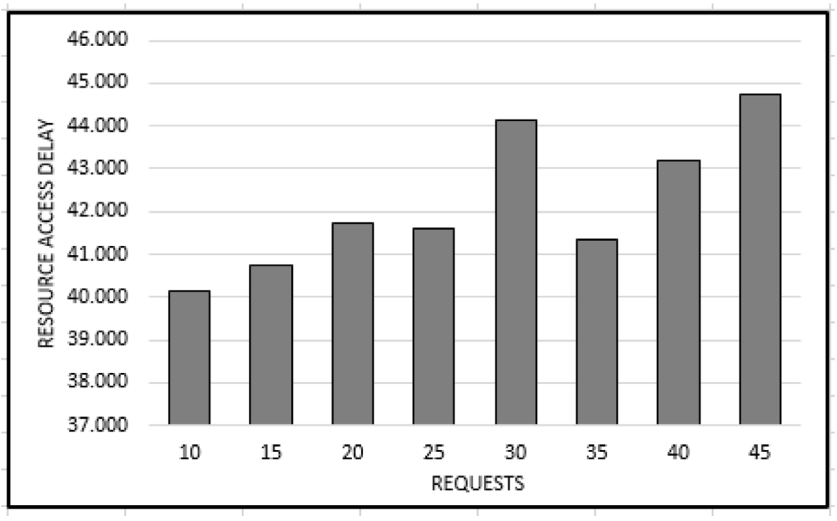

Fig. 39. Variation of resource access delay for 10 nodes when delay is 1 s, observed between node to node.

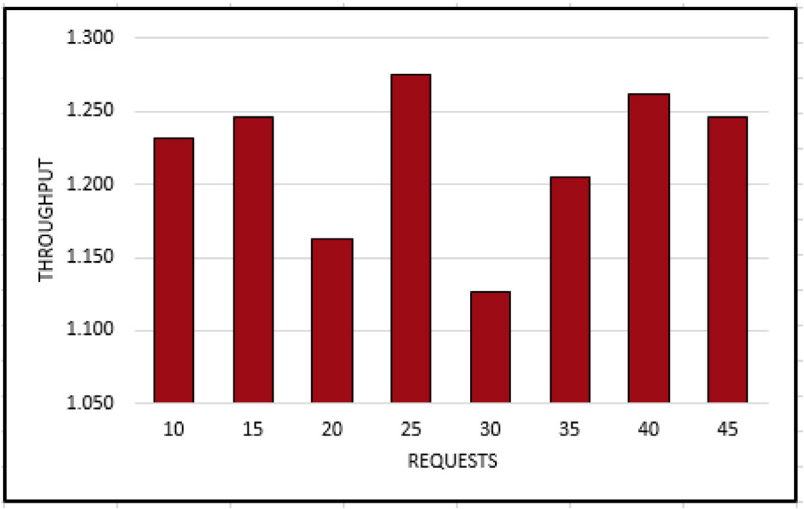

Fig. 40. Throughput for 10 nodes when delay is 1 s, observed between node to node.

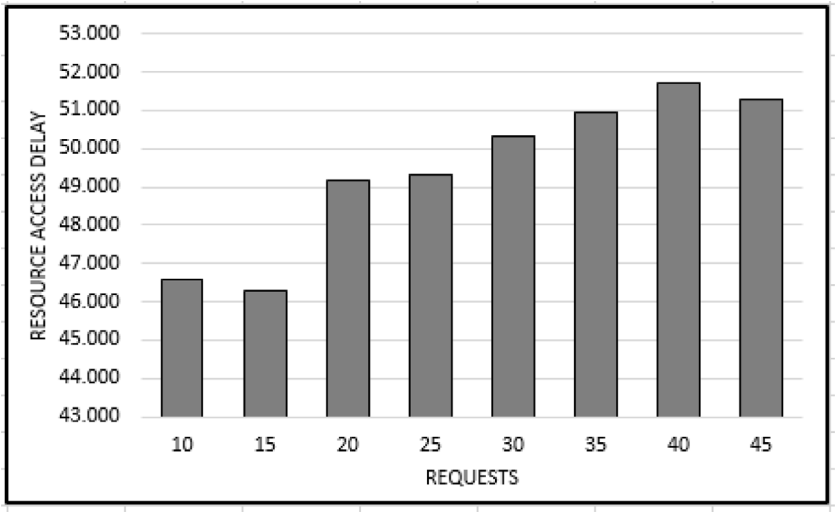

Fig. 41. Resource access delay for 10 nodes when delay is 5 s, observed between node to node.

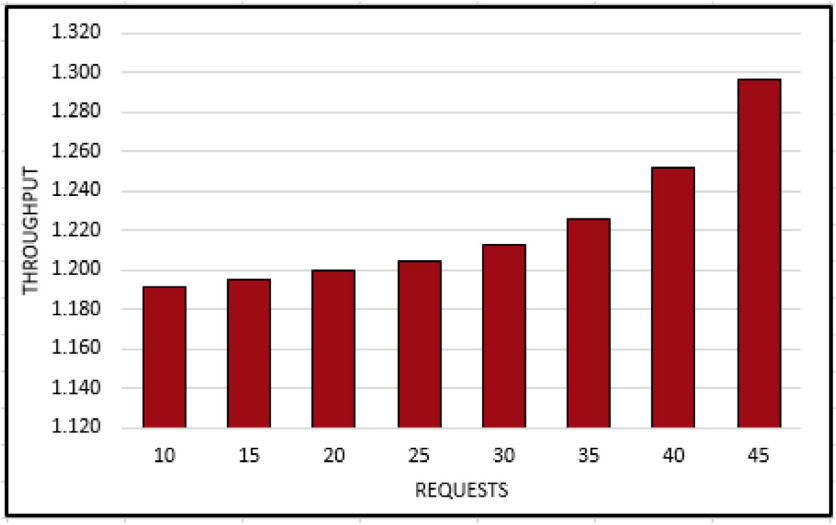

Fig. 42. Throughput for 10 nodes when delay is 5 s, observed between node to node.

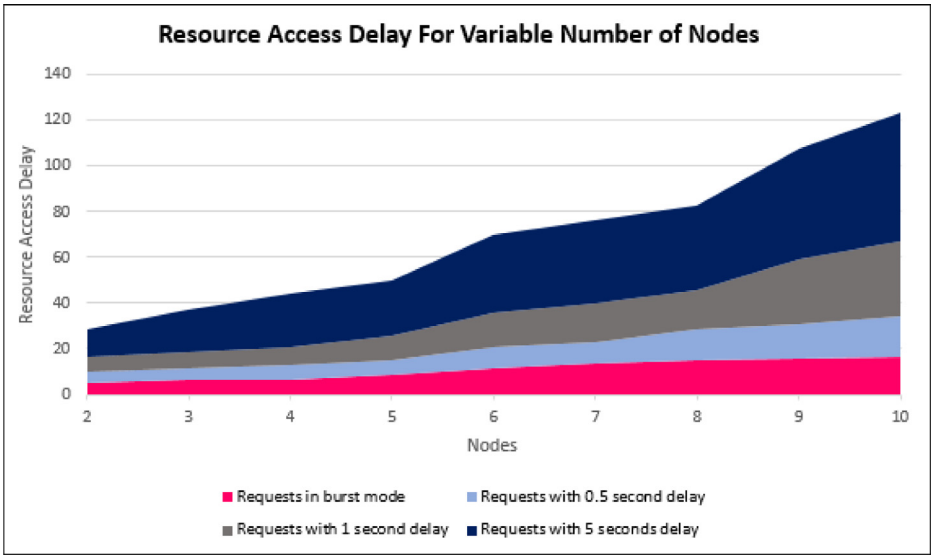

Fig. 43. Resource access delay for variable number of nodes.

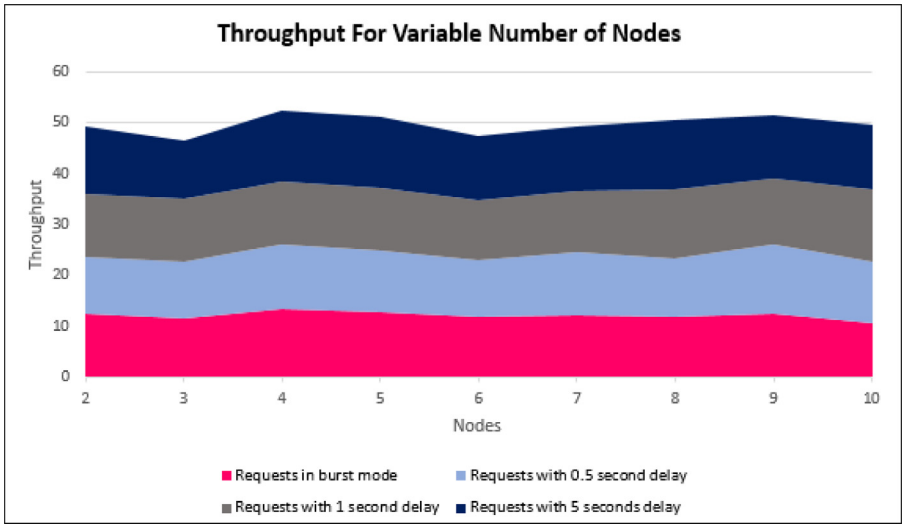

Fig. 44. Throughput for variable number of nodes.

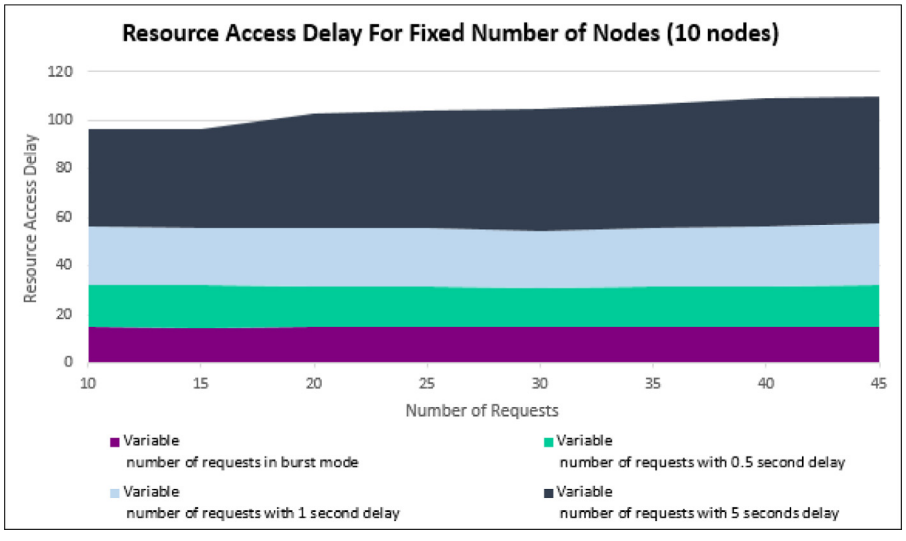

Fig. 45. Resource access delay for 10 constant number of nodes.

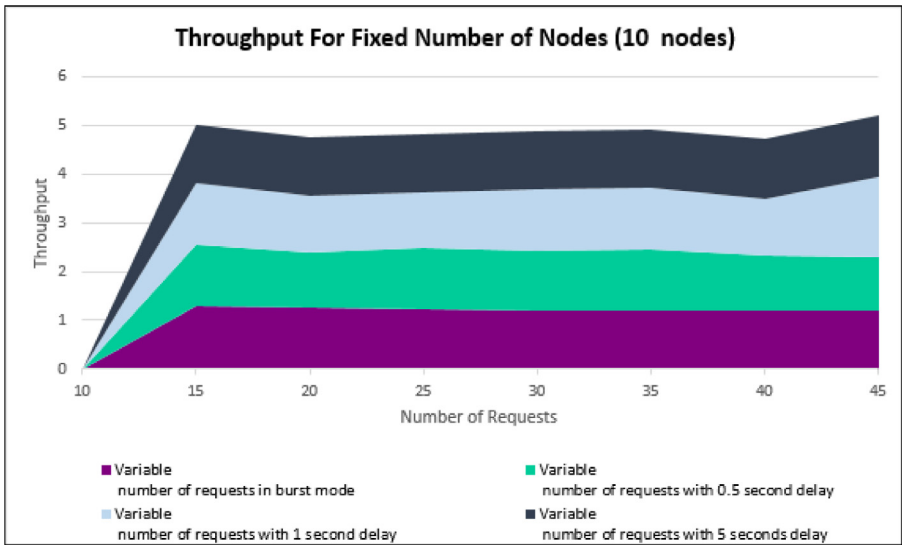

Fig. 46. Throughput for 10 constant number of nodes.

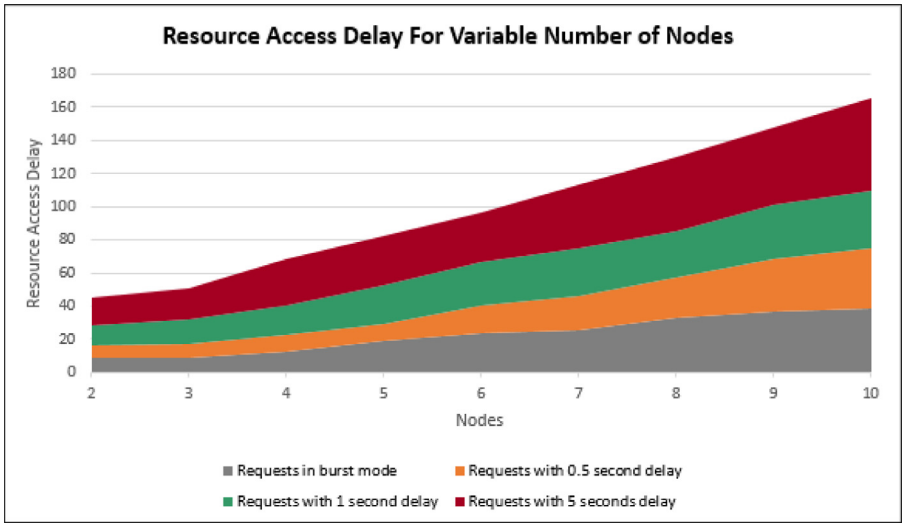

Fig. 47. Resource access delay for variable number of nodes.

have inconsequential impact on throughput as observed by graphical representation of Fig. 44. Similarly, Figs. 45 and 46 depicts the results of environment where the number of nodes are kept constant and number of requests are gradually increased. The graphical comparison reveals that impact of variable requests from fixed nodes does not radically effects resource access delay. While analyzing throughput, a consistent trend is observed after 15 requests.

6.3.2. End-to-end

Similarly, after considering the graphical results of end-to-end environment in Figs. 47 and 48 for variable number of nodes, it is perceived that increase in number of nodes certainly increases the resource access delay. However no significant change is observed in throughput. For constant number of 10 nodes with variable number of requests, resource access delay presented in Fig. 49 depicts undeviating trend.

In Fig. 50, fluctuation throughput values is observed when number of request is 15 and 0.5, 1 and 5 s delay is imposed on constant 10 nodes. Analysis of these results clarifies the performance bottlenecks which need to be managed and avoided congruently.

7. Conclusion

SD-IoT and blockchain are powerful combination causing significant impact across several IoT domains and can resolve security and privacy concerns that are rising due to growing usage of IoT devices. SD-IoT enables centralized management and monitoring of the IoT network by separating the control plane from data, and the forwarding plane from autonomous network devices. Dynamically programming and reorganizing network environments from the SD-IoT controller simplifies the network setup. With blockchain technology, trusted nodes involved in a network can track any data transaction. Because IoT devices have fewer resource capacities for performing smart activities, reliance on any third-party application or service provider to make choices about information collection, storage, and safety, is not an option. Digital security criteria for IoT environment such as accessibility, accountability, confidentiality and integrity is addressed by the development of blockchain technology. This paper presented a novel architecture that integrates SD-IoT with blockchain smart contracts for the implementation of immutable, verifiable, adaptive and automated access control policies and illustrates the usefulness of centralized and efficient access control for the IoT environment. After the evaluation of experiments,

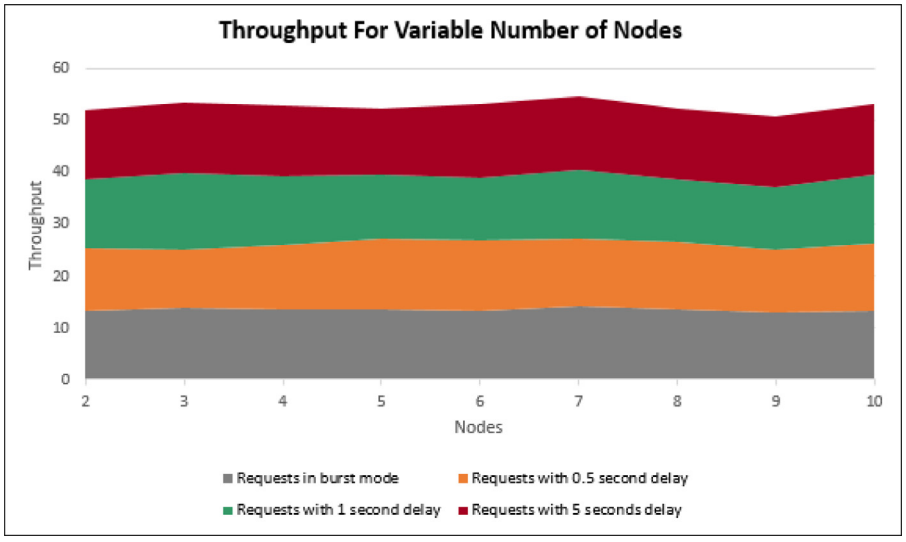

Fig. 48. Throughput for variable number of nodes.

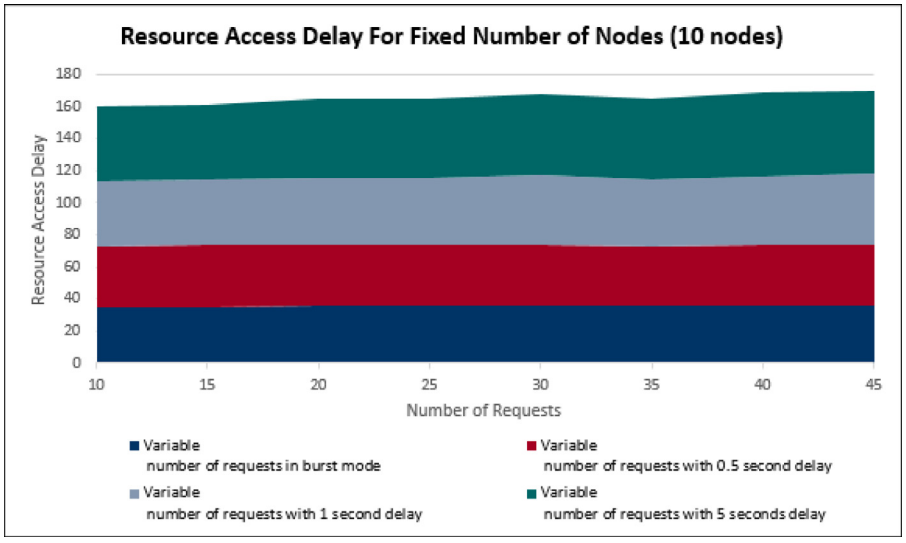

Fig. 49. Resource access delay for 10 constant number of nodes.

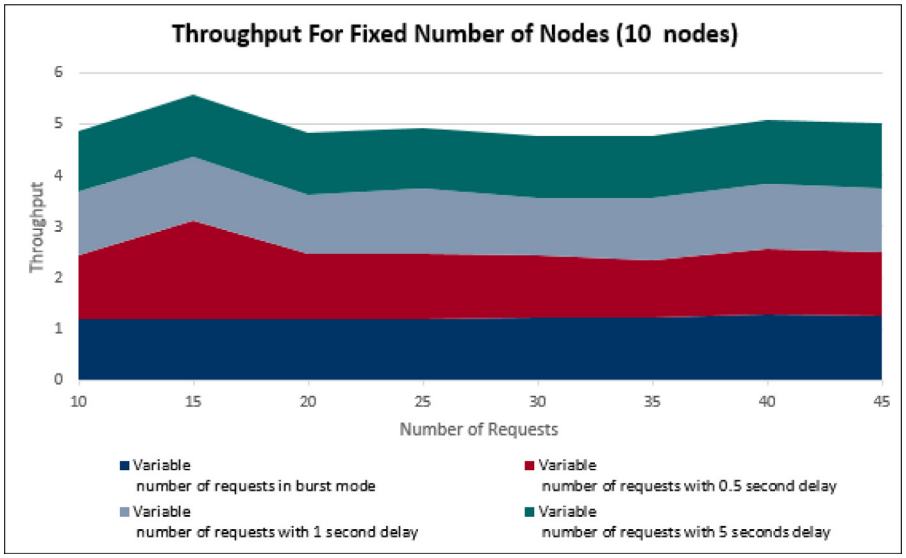

Fig. 50. Throughput for 10 constant number of nodes.

it is observed that the number of nodes predominates the number of requests. There is a minimal alteration for throughput when the number of nodes is fixed. Experiments for the variable number of requests for resource access created a minute shift in throughput and resource access delay. However, the sudden rise in the number of nodes caused a slight increase in the delay for resource access. By acknowledging these results preemptive performance degradation can be addressed accordingly. The proposed solution can be implemented in different domains with different security modules in future for analyzing the performance and upgrading the security of IoT networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] J. Sengupta, S. Ruj, S.D. Bit, A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT, J. Netw. Comput. Appl. 149 (2020) 102481.

[2] F. Li, Y. Han, C. Jin, Practical access control for sensor networks in the context of the internet of things, Comput. Commun. 89 (2016) 154–164.

[3] D. Hussein, E. Bertin, V. Frey, A community-driven access control approach in distributed IoT environments, IEEE Commun. Mag. 55 (3) (2017) 146–153.

[4] I. Ali, A.I.A. Ahmed, A. Almogren, M.A. Raza, S.A. Shah, A. Khan, A. Gani, Systematic literature review on IoT-based botnet attack, IEEE Access (2020).

[5] J. Bhayo, R. Jafaq, A. Ahmed, S. Hameed, S.A. Shah, A time-efficient approach toward ddos attack detection in IoT network using SDN, IEEE Internet Things J. 9 (5) (2022) 3612–3630, http://dx.doi.org/10.1109/JIOT.2021.3098029.

[6] F.I. Khan, S. Hameed, Software defined security service provisioning framework for internet of things, Int. J. Adv. Comput. Sci. Appl. 7 (12) (2016) http://dx.doi.org/10.14569/IJACSA.2016.071254, URL http://dx.doi.org/10.14569/IJACSA.2016.071254.

[7] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, J. Wan, Smart contract-based access control for the internet of things, IEEE Internet Things J. 6 (2) (2018) 1594–1605.

[8] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008, Available at: https://bitcoin.org/bitcoin.pdf.

[9] A. Narayanan, J. Clark, Bitcoin's academic pedigree, Commun. ACM 60 (12) (2017) 36–45.

[10] A. Narayanan, J. Clark, Bitcoin's academic pedigree, ACM Queue Mag. 15 (4) (2017) 1–30.

[11] J. Mendling, I. Weber, W. van der Aalst, et al., Blockchains for business process management – challenges and opportunities, ACM Trans. Manage. Inform. Syst. 9 (1) (2018) 1–16.

[12] M. Janssen, V. Weerakkody, E. Ismagilova, U. Sivarajah, Z. Irani, A framework for analysing blockchain technology adoption: Integrating institutional, market and technical factors, Int. J. Inf. Manage. 50 (2020) 302–309.

[13] A. Reyna, C. Martín, J. Chen, E. Soler, M. Díaz, On blockchain and its integration with IoT. Challenges and opportunities, Future Gener. Comput. Syst. 88 (2018) 173–190.

[14] S. Rouhani, R. Deters, Security, performance, and applications of smart contracts: A systematic survey, IEEE Access 7 (2019) 50759–50779.

[15] S. Ruj, A. Nayak, A decentralized security framework for data aggregation and access control in smart grids, IEEE Trans. Smart Grid 4 (1) (2013) 196–205.

[16] H. Cheung, C. Yang, H. Cheung, New smart-grid operation-based network access control, in: 2015 IEEE Energy Conversion Congress and Exposition, ECCE, IEEE, 2015, pp. 1203–1207.

[17] N. Tapas, G. Merlino, F. Longo, Blockchain-based IoT-cloud authorization and delegation, in: 2018 IEEE International Conference on Smart Computing, SMARTCOMP, IEEE, 2018, pp. 411–416.

[18] P.K. Sharma, J.H. Park, Blockchain based hybrid network architecture for the smart city, Future Gener. Comput. Syst. 86 (2018) 650–655.

[19] R.J. Robles, T.-h. Kim, D. Cook, S. Das, A review on security in smart home development, Int. J. Adv. Sci. Technol. 15 (2010).

[20] A. Dorri, S.S. Kanhere, R. Jurdak, P. Gauravaram, Blockchain for IoT security and privacy: The case study of a smart home, in: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), IEEE, 2017, pp. 618–623.

[21] A. FERREIRAabd, C.-C. Ricardo, L. Antunes, D. Chadwick, Access control: how can it improve patients' healthcare? Med. Care Compunet. 4 (4) (2007) 65.

[22] A. Sajid, H. Abbas, Data privacy in cloud-assisted healthcare systems: state of the art and future challenges, J. Med. Syst. 40 (6) (2016) 155.

[23] S. Qi, Y. Lu, W. Wei, X. Chen, Efficient data access control with fine-grained data protection in cloud-assisted IIoT, IEEE Internet Things J. 8 (4) (2020) 2886–2899.

[24] J. Sengupta, S. Ruj, S.D. Bit, A secure fog-based architecture for industrial Internet of Things and industry 4.0, IEEE Trans. Ind. Inform. 17 (4) (2020) 2316–2324.

[25] A. Kumari, R. Gupta, S. Tanwar, N. Kumar, A taxonomy of blockchain-enabled softwarization for secure UAV network, Comput. Commun. 161 (2020) 304–323.

[26] S. Siddiqui, S. Hameed, S.A. Shah, I. Ahmad, A. Aneiba, D. Draheim, S. Dustdar, Toward software-defined networking-based IoT frameworks: A systematic literature review, taxonomy, open challenges and prospects, IEEE Access 10 (2022) 70850–70901, http://dx.doi.org/10.1109/ACCESS.2022.3188311.

[27] S. Banerjee, B. Bera, A.K. Das, S. Chattopadhyay, M.K. Khan, J.J. Rodrigues, Private blockchain-envisioned multi-authority CP-ABE-based user access control scheme in IIoT, Comput. Commun. 169 (2021) 99–113, http://dx.doi.org/10.1016/j.comcom.2021.01.023.

[28] C. Tselios, I. Politis, S. Kotsopoulos, Enhancing SDN security for IoT-related deployments through blockchain, in: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN, IEEE, 2017, pp. 303–308.

[29] N. Szabo, Formalizing and securing relationships on public networks, First Monday 2 (9) (1997).

[30] M.S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, M.H. Rehmani, Applications of blockchains in the Internet of Things: A comprehensive survey, IEEE Commun. Surv. Tutor. 21 (2) (2018) 1676–1717.

[31] M.A. Khan, K. Salah, IoT security: Review, blockchain solutions, and open challenges, Future Gener. Comput. Syst. 82 (2018) 395–411.

[32] N. Kshetri, Can blockchain strengthen the internet of things? IT Prof. 19 (4) (2017) 68–72.

[33] B.A.A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, A survey of software-defined networking: Past, present, and future of programmable networks, IEEE Commun. Surv. Tutor. 16 (3) (2014) 1617–1634.

[34] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, OpenFlow: Enabling innovation in campus networks, SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74.

[35] R. Chaudhary, N. Kumar, LOADS: Load optimization and anomaly detection scheme for software-defined networks, IEEE Trans. Veh. Technol. 68 (12) (2019) 12329–12344.

[36] R.N.B. Rais, M. Mendonca, T. Turletti, K. Obraczka, Towards truly heterogeneous internets: Bridging infrastructure-based and infrastructure-less networks, in: 2011 Third International Conference on Communication Systems and Networks, COMSNETS 2011, IEEE, 2011, pp. 1–10.

[37] C. Vandana, Security improvement in iot based on software defined networking (sdn), Int. J. Sci., Eng. Technol. Res. 5 (1) (2016) 2327–4662.

[38] J. Bhayo, S. Hameed, S.A. Shah, An efficient counter-based ddos attack detection framework leveraging software defined IoT (SD-IoT), IEEE Access 8 (2020) 221612–221631.

[39] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, N. Venkatasubramanian, A software defined networking architecture for the internet-of-things, in: 2014 IEEE Network Operations and Management Symposium, NOMS, IEEE, 2014, pp. 1–9.

[40] K. Sood, S. Yu, Y. Xiang, Software-defined wireless networking opportunities and challenges for internet-of-things: A review, IEEE Internet Things J. 3 (4) (2015) 453–463.

[41] D. Hussein, E. Bertin, V. Frey, Access control in IoT: From requirements to a candidate vision, in: 2017 20th Conference on Innovations in Clouds, Internet and Networks, ICIN, IEEE, 2017, pp. 328–330.

[42] K. Toyoda, K. Machi, Y. Ohtake, A.N. Zhang, Function-level bottleneck analysis of private proof-of-authority ethereum blockchain, IEEE Access 8 (2020) 141611–141621.

[43] M.A. Manolache, S. Manolache, N. Tapus, Decision making using the blockchain proof of authority consensus, Procedia Comput. Sci. 199 (2022) 580–588.

[44] B.N. Silva, M. Khan, K. Han, Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities, Sustainable Cities Soc. 38 (2018) 697–713.

[45] A. Ouaddah, I. Bouij-Pasquier, A.A. Elkalam, A.A. Ouahman, Security analysis and proposal of new access control model in the internet of thing, in: 2015 International Conference on Electrical and Information Technologies, ICEIT, IEEE, 2015, pp. 30–35.

[46] A. Ouaddah, A blockchain based access control framework for the security and privacy of IoT with strong anonymity unlinkability and intractability guarantees, in: Advances in Computers, vol. 115, Elsevier, 2019, pp. 211–258.

[47] B. Chen, Y.-L. Huang, M. Güneş, S-CBAC: A secure access control model supporting group access for internet of things, in: 2015 IEEE International Symposium on Software Reliability Engineering Workshops, ISSREW, IEEE, 2015, 67–67.

[48] N. Ye, Y. Zhu, R.-c. Wang, R. Malekian, L. Qiao-Min, An efficient authentication and access control scheme for perception layer of internet of things, Appl. Math. Inform. Sci. 8 (4) (2014) 1617.

[49] S. Aggarwal, R. Chaudhary, G.S. Aujla, A. Jindal, A. Dua, N. Kumar, Energychain: Enabling energy trading for smart homes using blockchains in smart grid ecosystem, in: Proceedings of the 1st ACM MobiHoc Workshop on Networking and Cybersecurity for Smart Cities, 2018, pp. 1–6.

[50] O. Novo, Blockchain meets IoT: An architecture for scalable access management in IoT, IEEE Internet Things J. 5 (2) (2018) 1184–1195.

[51] M.U. Rahman, B. Guidi, F. Baiardi, Blockchain-based access control management for decentralized online social networks, J. Parallel Distrib. Comput. 144 (2020) 41–54.
[52] D.D.F. Maesa, P. Mori, L. Ricci, A blockchain based approach for the definition of auditable access control systems, Comput. Secur. 84 (2019) 93–119.
[53] A. Dorri, S.S. Kanhere, R. Jurdak, Towards an optimized blockchain for IoT, in: Proceedings of the Second International Conference on Internet-of-Things Design and Implementation, ACM, 2017, pp. 173–178.
[54] R. Xu, Y. Chen, E. Blasch, G. Chen, Blendcac: A smart contract enabled decentralized capability-based access control mechanism for the iot, Computers 7 (3) (2018) 39.
[55] O. Alphand, M. Amoretti, T. Claeys, S. Dall'Asta, A. Duda, G. Ferrari, F. Rousseau, B. Tourancheau, L. Veltri, F. Zanichelli, IoTChain: A blockchain security architecture for the internet of things, in: 2018 IEEE Wireless Communications and Networking Conference, WCNC, IEEE, 2018, pp. 1–6.
[56] S. Hameed, S.A. Shah, Q.S. Saeed, S. Siddiqui, I. Ali, A. Vedeshin, D. Draheim, A scalable key and trust management solution for IoT sensors using SDN and blockchain technology, IEEE Sens. J. (2021) http://dx.doi.org/10.1109/JSEN.2021.3052009, 1–1.
[57] R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, K. Salah, A user authentication scheme of iot devices using blockchain-enabled fog nodes, in: 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications, AICCSA, IEEE, 2018, pp. 1–8.
[58] G. Zyskind, O. Nathan, et al., Decentralizing privacy: Using blockchain to protect personal data, in: 2015 IEEE Security and Privacy Workshops, IEEE, 2015, pp. 180–184.
[59] L. Mendiboure, M.A. Chalouf, F. Krief, A scalable blockchain-based approach for authentication and access control in software defined vehicular networks, in: 2020 29th International Conference on Computer Communications and Networks, ICCCN, IEEE, 2020, pp. 1–11.
[60] A.G. Abbasi, Z. Khan, Veidblock: Verifiable identity using blockchain and ledger in a software defined network, in: Companion Proceedings of The10th International Conference on Utility and Cloud Computing, ACM, 2017, pp. 173–179.
[61] P.K. Sharma, S. Singh, Y.-S. Jeong, J.H. Park, Distblocknet: A distributed blockchains-based secure sdn architecture for iot networks, IEEE Commun. Mag. 55 (9) (2017) 78–85.
[62] A. Ouaddah, H. Mousannif, A.A. Elkalam, A.A. Ouahman, Access control in the Internet of Things: Big challenges and new opportunities, Comput. Netw. 112 (2017) 237–262, http://dx.doi.org/10.1016/j.comnet.2016.11.007.
[63] T.M. Fernández-Caramés, P. Fraga-Lamas, A review on the use of blockchain for the Internet of Things, IEEE Access 6 (2018) 32979–33001.
[64] K. Christidis, M. Devetsikiotis, Blockchains and smart contracts for the internet of things, Ieee Access 4 (2016) 2292–2303.
[65] A. Ouaddah, A. Abou Elkalam, A. Ait Ouahman, FairAccess: a new blockchain-based access control framework for the Internet of Things, Secur. Commun. Netw. 9 (18) (2016) 5943–5964.
[66] A. Ouaddah, A.A. Elkalam, A.A. Ouahman, Towards a novel privacy-preserving access control model based on blockchain technology in IoT, in: Europe and MENA Cooperation Advances in Information and Communication Technologies, Springer, 2017, pp. 523–533.
[67] M. Steichen, S. Hommes, R. State, ChainGuard—A firewall for blockchain applications using SDN with OpenFlow, in: 2017 Principles, Systems and Applications of IP Telecommunications, IPTComm, IEEE, 2017, pp. 1–8.
[68] K.-P. Yu, L. Tan, M. Aloqaily, H. Yang, Y. Jararweh, Blockchain-enhanced data sharing with traceable and direct revocation in IIoT, IEEE Trans. Ind. Inform. (2021).
[69] J. Polge, J. Robert, Y. Le Traon, Permissioned blockchain frameworks in the industry: A comparison, Ict Express 7 (2) (2021) 229–233.
[70] Truffle - trufflesuite/truffle: A tool for developing smart contracts, https://github.com/trufflesuite/truffle (Accessed on 21st March 2020).
[71] Ganache, https://www.trufflesuite.com/ganache (Accessed on 21st March 2020).

Mizna Khalid received the MS degree in Computer Network and Security from National University of Computer and Emerging Sciences, Pakistan. Prior to joining MS, she received BS degree in Computer Science from Jinnah University for Women, Pakistan. Her research interests include network security, internet of things, software defined network and blockchain.

Sufian Hameed received the Ph.D. degree in networks and information security from University of Göttingen, Germany. He works as Associate Professor at Department of Computer Science at National University of Computer and Emerging Sciences, Pakistan. He also leads the IT Security Labs at 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 IoTs.

Abdul Qadir received the BS degree in Computer Science from National University of Computer and Emerging Sciences, Pakistan. His research interests include network security, web security, data privacy and system designs revolving around these methodologies. He is currently working as a software development engineer at Securiti.ai, one of the leading data privacy companies.

Syed Attique Shah received the Ph.D. degree from the Institute of Informatics, Istanbul Technical University, Istanbul, Turkey. During his Ph.D. degree, he studied as 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 has worked as an Associate Professor and the Chairperson at the Department of Computer Science, BUITEMS, Quetta, Pakistan. He was also engaged as a Lecturer at the Data Systems Group, Institute of Computer Science, University of Tartu, Estonia. Currently, he is working as a Lecturer in Smart Computer Systems, at the School of Computing and Digital Technology, Birmingham City University, United Kingdom. His research interests include big data analytics, the Internet of Things, network security, and information management.

Dirk Draheim received the Ph.D. degree from Freie Universität Berlin and habilitation degree from Universität Mannheim, Germany. Currently, he is full professor of information systems and the head of the information systems group of 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.