

Distributed Attribute-Based Access Control System Using a Permissioned Blockchain

Sara Rouhani · Rafael Belchior · Rui S. Cruz · Ralph Deters

Received: date / Accepted: date

Abstract Auditing provides an essential security control in computer systems, by keeping track of all access attempts, including both legitimate and illegal access attempts. This phase can be useful to the context of audits, where eventual misbehaving parties can be held accountable. Blockchain technology can provide trusted auditability required for access control systems. In this paper, we propose a distributed Attribute-Based Access Control (ABAC) system based on blockchain to provide trusted auditing of access attempts. Besides auditability, our system presents a level of transparency that both access requestors and resource owners can benefit from it. We present a system architecture with an implementation based on Hyperledger Fabric, achieving high efficiency and low computational overhead. The proposed solution is validated through a use case of independent digital libraries. Detailed performance analysis of our implementation is presented, taking into account different consensus mechanisms and databases. The experimental evaluation shows that our presented system can process 5,000 access control requests with the send rate of 200 per second and a latency of 0.3 seconds.

Keywords Distributed Access Control · Attribute-Based Access Control · Blockchain · Hyperledger Fabric · Performance

Sara Rouhani, Ralph Deters
Department of Computer Science, University of Saskatchewan, Saskatoon, SK S7N5C9, Canada
E-mail: sara.rouhani@usask.ca
E-mail: deters@cs.usask.ca

Rafael Belchior, Rui S. Cruz
Department of Computer Science and Engineering, Instituto Superior Técnico, Universidade de Lisboa, Portugal
E-mail: rafael.belchior@tecnico.ulisboa.pt
E-mail: rui.s.cruz@tecnico.ulisboa.pt

1 Introduction

Access control systems exist to protect system resources from unauthorized accesses. Based on the system policies, security procedures within the organization, and the level of the sensitivity of the resources, the access control systems follow one of the available access control models.

Attribute-Based Access Control (ABAC) [22,54] is an access control model that regulates access permissions, based on the characteristics (in this context called attributes) of subjects, resources, and context (or environment). Access decisions are made by evaluating these attributes based on defined policies. Fig. 1 shows the overview of the ABAC model.

ABAC has some advantages over other access control models as, (a) it can provide fine-grained and flexible access control because it allows an arbitrary number of attributes in access control decisions; (b) the implementation of complex policies is simple and applicable; and (c) it can provide dynamic and effective access control decisions by involving environmental attributes in decision making.

Auditing is one of the essential controls in systems security. Auditing is the action of tracking all access attempts, including both legitimate and illegal access attempts. Keeping track of legitimate access attempts helps with non-repudiation, and keeping track of illegal access attempts helps with identifying potential threats. Auditability is also one of the key characteristics of blockchain by providing a trustable history of traceable transactions [7,6]. Blockchain can exploit smart contracts to store access control policies, process access decisions, store the result of access decisions, and accountability regarding stakeholders with different incentives. Then, at any point in the future, all access

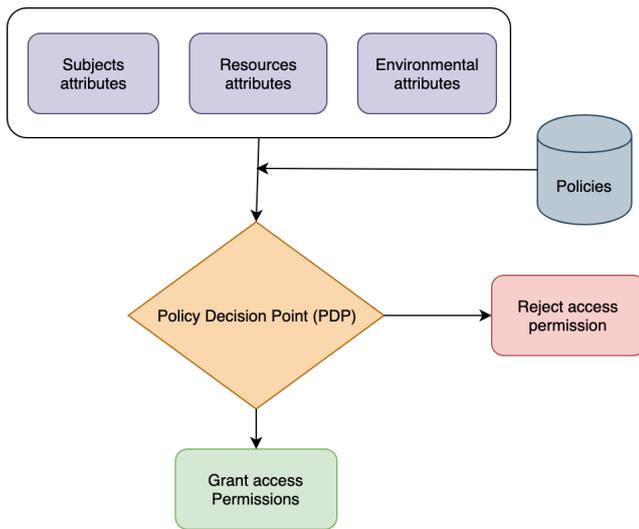


Fig. 1 Attribute based access control.

attempts toward a particular resource can be queried from the blockchain. This feature can be used as an authentic proof for non-repudiation, or it can be studied for further analysis to identify possible threats.

Besides, blockchain presents other beneficial features that are desirable for access control systems such as immutability and transparency. For example, if a malicious system administrator changes a policy to grant or deny someone access, it will be recorded on the blockchain, and it is not possible to delete the trace of updates on policies from the blockchain. For each policy, all history of changes applied in the policy can be queried by permissioned users in permissioned blockchain. However, we prevent such a problem by configuring smart contracts so that authenticated parties must approve any change in access control policies before execution.

In this study, we propose a complete end to end solution for implementing an ABAC system with the focus on policy-based architecture based on Hyperledger Fabric permissioned blockchain¹. Our paper contributions are summarized as follows:

- We propose an architecture for implementing a flexible access control system based on ABAC and permissioned blockchain
- We discuss our access control components including Policy Information Point (PIP), Policy Decision Point (PDP), Policy Administration Point (PAP), which are implemented as smart contracts (*or Chain-code*)
- We provide a specific use case of digital libraries to represent the system operation modelling.
- We carried out experiments, and we analyzed the performance of the presented access control appli-

cation using Hyperledger Caliper² based on multiple configurations, including different databases and consensus methods.

- Our performance analysis results also conduct a comprehensive comparison between various network configurations and pluggable components in Hyperledger Fabric modular architecture.

The remainder of this paper is organized as follows. The initial concepts of blockchain and access control systems are presented in Section 2, followed by Section 3, which reviews related studies. Section 4 explains the system model, architecture and representation of designed components. A case study based on access to the digital libraries' resources is introduced in Section 5. Section 6 presents the evaluation results based on the represented case study and multiple configurations. Finally, Section 7 concludes the paper and refers to our future work.

2 Background

2.1 Blockchain and smart contracts

Blockchain is a particular type of distributed ledger technology. The data is recorded on the blockchain as a group of transactions called blocks. Each block has a hash value, and it links to the previous block by referencing the hash value of the previous block in the header of the current block. Consequently, data manipulation is not possible in the blockchain, as even a slight change leads to an inconsistency between linked blocks, and can be recognized easily. In order to attach a valid block to the blockchain, a consensus mechanism is applied. There are several consensus mechanisms with a trade-off between performance and security.

Before the development of smart contracts, blockchain applications were limited to creating cryptocurrencies and simple monetary transactions. The development of smart contracts has provided the infrastructure for creating more diverse blockchain-based applications. Smart contracts are executable logic encoded in blockchain with the ability to enforce automatically.

Blockchain networks can be divided into two main categories: public and permissioned.

Public blockchains are open to the world, and every user can join the blockchain with an anonymous identity, submit a transaction, and participate in consensus. Permissioned blockchains include an additional membership layer, so only authenticated users can join the blockchain and interact with different components.

¹ <https://www.hyperledger.org/projects/fabric>

² <https://www.hyperledger.org/projects/caliper>

Today many blockchain platforms exist, and they are geared toward implementing smart contracts and decentralized applications. They are different in different aspects, such as the type of the network (public or permissioned), built-in cryptocurrency, transaction workflow, performance, privacy, cost and, most importantly, maturity. Some blockchain platforms such as Ethereum, Hyperledger Fabric, Corda have mature tools, while others have very little support for their users and developers.

Hyperledger Fabric [4] is a famous implementation of a permissioned blockchain, hosted by the Linux Foundation. Hyperledger Fabric has a modular structure that allows component pluggability, such as consensus, membership, and database. The membership layer can authenticate users and grant access to users based on their access level and system policy. Hyperledger Fabric has integrated the ABAC mechanism, so it is possible to build permission groups for access control by checking members' attributes. However, access control parameters and permission groups have to be predefined. It is not suitable for applications that require dynamic and flexible access control.

Our presented system provides a solution for off-chain parties that look for a flexible and distributed access control service compatible with their authentication service. Our provided solution can be easily integrated with any off-chain system, while access control functionality is achieved through blockchain and smart contracts.

Smart contracts correspond to logic encoded in the blockchain that can be programmed and deployed as an automation program. Accordingly, they can create complex transactions and enforce their conditions automatically [45].

Chaincode is a term introduced by Hyperledger Fabric for smart contracts. Chaincode may consist of multiple smart contracts or include only one smart contract. We use the chaincode and smart contract concepts interchangeably.

Before the development of smart contracts, blockchain applications were limited in creating cryptocurrencies and simple monetary transactions. The development of smart contracts provided the infrastructure for creating more diverse blockchain-based applications, such as healthcare [26, 5, 11], Internet of Things (IoT) [25, 33], resource sharing [58, 49], and business process management [51, 28, 41]. In our previous paper [44], we discussed that although the applications of these systems are different, their primary purpose is similar as they aim to control access over particular data. The domain of the data is their main difference; for example, it could

be patient healthcare data or data generated by IoT devices.

2.2 Access control models and ABAC

Access control refers to any action to prevent data and resources from unauthorized access, disclosure or modification. In traditional databases, an authorization defined by a triplet $\langle o, s, p \rangle$ and defines that subject s is authorized to execute privilege p on object o [9].

Conventional access control models follow such definition, adding into consideration the context in which an access control request is performed.

(1) Discretionary Access Control (DAC) [10] or authorization-based, (2) Mandatory Access Control (MAC) [8], and later (3) Role-Based Access Control (RBAC) [17] are three initial access control models [47].

DAC restricts access permissions based on the subjects' identity, and the resource owner defines policy rules.

In MAC, the system defines access policies through the security labels. This model usually is used for controlling access over sensitive and confidential data.

In RBAC, there are predefined roles in the system and users have different access levels depending on their roles.

ABAC is logical access control that comprises access control lists, role-based access control, and its own method for providing access based on the evaluation of attributes [22]. ABAC controls access to the system resources by evaluating policies (system rules) against entities' attributes, including subject, object, and environmental attributes. Attributes are characteristics of the subjects (users) and protected objects (resources). The environment conditions as the environment's attributes can also be taken into account for ABAC decision making.

ABAC is a flexible and fine-grained mechanism that is also capable of enforcing the other three methods. Distributed systems also adopted ABAC as they require federation and autonomy control among coordinated systems, and ABAC enables granular and meta attribute capabilities that support privilege delegation in a distributed application [23]. Likewise, blockchain-based applications mostly adopt ABAC [44].

XACML [3] introduces a policy-based architecture for the specification and enforcement of access control policies. The architecture comprises the following components.

- *Client*: the device that requests access to a resource, possibly on behalf of a user.

- *Policy Enforcement Point (PEP)*: the network device on which access decisions are carried out. PEP serves as the gatekeeper to the intended resource.
- *Policy Information Point (PIP)*: the repository that holds information (attributes) about the client and provides this information to the PDP.
- *Policy Decision Point (PDP)*: the component that decides to allow or deny the client access to the resource.
- *Policy Administration Point (PAP)*: the component that is responsible for managing access control policies.
- *Accounting or Auditing*: The component that is responsible for tracking access attempts.

Figure 2 illustrates how these components interact with the client and each other. A client requests access permission, and Policy Enforcement Point (PEP) forwards the request to Policy Decision Point (PDP). Policy Decision Point (PDP) queries the related policy and attributes from Policy Administration Point (PAP) and Policy Information Point (PIP). After receiving the required information, Policy Decision Point (PDP) assesses the access decision and sends the result to Policy Enforcement Point (PEP) for enforcing the access decision.

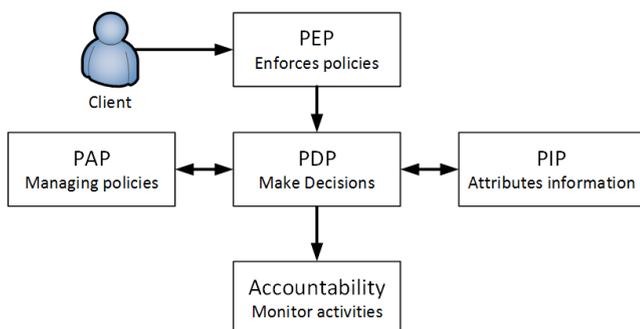


Fig. 2 ABAC logical components based on Policy based architecture [3]

3 Related Work

Many studies on blockchain technology focus on presenting an access control system either in the context of specific applications, such as healthcare [55, 43, 5, 52, 11, 42], IoT [34, 57, 14, 39, 35, 13, 30], and cloud federation [16, 2] or they introduce a general access control system, which can be employed for different applications. In our previous study [44], we have investigated the state of the art of access control systems based on blockchain. In this section, we overview similar studies,

which present an attribute-based access control based on blockchain. As illustrated in table Section 3, many studies use the attributed-based method for their access control system because of the granularity, flexibility, and dynamic features that ABAC provides.

Guo *et al.* in [19] introduce a hybrid architecture for access control over Electronic Health Record (EHR) data using blockchain and edge nodes. The blockchain acts as a tamper-proof validation component to verify identities and access control policies. The edge nodes store the EHR data off-chain and enforce the access controls. The smart contracts include the address of EHR data on the edge nodes by using one-time self-destructing URLs³. Based on performance results against unauthorized retrieval for the average transaction processing time was 40 ms, and the average response time was 30 ms. Also, the test result based on a high number of patients does not affect the response time and indicate the scalability of their solution.

Zyskind *et al.* in [60] conceptualizes the blockchain technology as an access control moderator, complemented by an off-blockchain storage solution. Blockchain clients representing users that provide their data to a service provider are the owners of their data. Based on that premise, this solution is meant to empower users, so they have the information about which data is collected about them by third parties and how their data is used. For achieving that goal, each data owner can issue transactions, used to change the set of permissions granted to a service or entity. Each transaction is recorded on the blockchain, allowing for auditability and traceability.

Zhang *et al.* in [57] propose a solution directed to IoT blockchain-based access control. The authors introduce the concepts of *Judge Contract (JC)*, *Register Contract (RC)* and *Access Control Contracts (ACC)*. Access control contracts store access control policies for a subject-object pair. In this system, both JC and RC are essential pieces in regards to achieving a distributed and reliable access control. The JC receives misbehaviour reports and applies penalties according to them. The RC stores the misbehaviour information from the JC and manages it through the judging method. Moreover, it stores information such as name, subject, object, and smart contract for access control.

Zhu *et al.* in [59] propose a transaction based access control (TBAC) system that integrates ABAC model into the bitcoin blockchain. There are four implemented transactions, including subject registration, object escrowing and publication, access request and grant. They also present a cryptosystem associated with their system as an additional security layer. The system is eval-

³ <https://1ty.me/>

Table 1 A summary of blockchain-based access control applications.

Research paper	Domain	Access control method	Privacy Support	Scalability
Jemel and Serhrouchni [24]	Data sharing	ABAC + Attribute-based Encryption	✓	x
Wang <i>et al.</i> [50]	Data sharing	ABAC + Attribute-based Encryption	✓	x
Zhu <i>et al.</i> [59]	Resource sharing	ABAC	✓	x
Hu <i>et al.</i> [21]	Knowledge sharing	Fine-grained	x	x
Ferdous <i>et al.</i> [16]	Cloud federation	-	x	x
Alansari <i>et al.</i> [2]	Cloud federation	ABAC	✓	x
Zhang and Posland [55]	Health care	ABAC	✓	x
Rouhani <i>et al.</i> [43]	Health care	Role-based	✓	✓
Guo <i>et al.</i> [19]	Health care	ABAC	x	✓
Asaph <i>et al.</i> [5]	Health care	-	✓	✓
Xia <i>et al.</i> [52]	Health care	-	✓	x
Dagher <i>et al.</i> [11]	Health care	Role-based	✓	✓
Rajput <i>et al.</i> [42]	Health care	Role-based	✓	✓
Zyskind <i>et al.</i> [60]	Mobile applications	Policy-based	✓	✓
Novo [34]	IoT	-	x	✓
Outchakoucht <i>et al.</i> [36]	IoT	Policy-based	x	x
Zhang <i>et al.</i> [57]	IoT	Policy-based and dynamic access control	x	x
Dukkipati <i>et al.</i> [14]	IoT	ABAC	✓	x
Pinno <i>et al.</i> [40]	IoT	ABAC	✓	✓
Ouaddah <i>et al.</i> [35]	IoT	-	✓	✓
Ding <i>et al.</i> [13]	IoT	ABAC	x	✓
Ma <i>et al.</i> [30]	IoT	Generic	✓	✓
Rouhani <i>et al.</i> [46]	Physical access control	Role-based	x	✓
Es-Samaali <i>et al.</i> [15]	Big data management	ABAC	✓	✓
Xh <i>et al.</i> [53]	Space situation awareness	Capability-Based	x	✓
Lyu <i>et al.</i> [29]	Information centric networking	matching-based	✓	x
Paillisse <i>et al.</i> [37]	Multi-administrative domain	-	✓	x
Laurent <i>et al.</i> [32]	General access control	Access Control List	✓	x
Maesa <i>et al.</i> [12]	General access control	ABAC	x	✓
Guo <i>et al.</i> [20]	General access control	ABAC	✓	x
Lee <i>et al.</i> [27]	General access control	Role-based	✓	x

uated in terms of security, but the performance and scalability of the system are not examined.

In the federated cloud services, access control enforcement is still vulnerable to privacy violations.

Alansari *et al.* in [2] present an attribute-based access control system based on Pedersen commitment scheme [38] and blockchain. The system is designed to keep the users' attributes private from the federated organization. Users' identity attributes and access control policies are stored on the blockchain to guarantee the integrity of them. They also employed Trusted hardware technology to guarantee the integrity of the policy enforcement process.

Zhang and Posland in [55] propose an architecture for granular access authorization that supports flexible queries, which provides secure authorization at different levels of granularity. The designed architecture offers a capable infrastructure without requiring the public key infrastructure (PKI), so it decreases the computation time needed and suitable for the devices with limited resources in EMR systems. As a result, their system can efficiently respond to a requester without exposing unauthorized private data.

Maesa *et al.* in [12] implemented an access control service on top of Ethereum ⁴. Blockchain is used to

store smart contracts that represent access control policies represented in eXtensible Access Control Markup Language (XACML) [3] and to perform the decision process. Such smart contracts are called *Smart Policies (SPs)*. Thus, SPs are responsible for the policy evaluation process, embedding a PDP for a specific access control policy. Each time, an access request needs to be evaluated to make an access decision, and the blockchain executes it in a distributed way. The decision is made based on information concerning the users. For this purpose, the concept of Attribute Manager (AM) is introduced. AMs are the components that manage the attributes of the entities involved in the process, such as subjects, resources, and environmental context. AMs can update and retrieve their values and are created by an entity, the Attribute Provider (AP). Implementation based on Ethereum blockchain is costly since, for every operation, a fee called gas must be paid. Although for public blockchain systems, it is generally unavoidable, for permissioned implementation, this is an unnecessary additional cost imposed on the system.

The privacy and security of the data generated by IoT devices are the major concern of the IoT system due to the extensive scale and distributed nature of IoT networks. In order to protect users' privacy, many studies have considered blockchain to provide secure access

⁴ www.ethereum.org

control to the IoT data. References [14, 40, 13] presented an ABAC for IoT systems. Dukkupati *et al.* solutions [14] store system policies off-chain while [40] stores the policies on the Ethereum platform and is less vulnerable to security breaches, but more costly. Ding *et al.* in [13] focus on simplifying the access control protocol to make it lightweight and suitable for IoT devices with limited computing capability and energy resources.

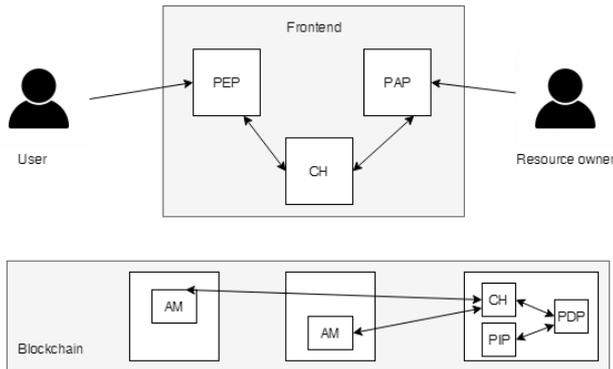


Fig. 3 Blockchain-based access control service architecture [31].

Zhang and Posland in [56] present an architecture for a blockchain-based Electronic Medical Record (EMR) access with granularity control that supports flexible data queries. The user layer first sends a query that could have different levels of granularity, such as block query, attribute query, or mixed query. The agent layer aggregates the query data and authorizes the user, who has access permission for that query. If it is a valid query, it passes the query to the storage layer. The storage layer returns the data to the agent layer. The agent layer encrypts the query and sends it to the user layer. Lastly, the query is decrypted, using the provided access keys. The represented architecture provides a flexible infrastructure to achieve granular access control without requiring Public Key Infrastructure (PKI), so it leads to a decrease in computation time proceeding.

As we reviewed, many studies have investigated blockchain as a back-end infrastructure for the distributed access control system. However, most of the prior works in this area are domain-specific. It means their access control solutions are designed for a particular domain, such as healthcare data or IoT data. Besides, most of these studies lack the details of implementation and performance analysis. As a result, it remains unclear if a blockchain can be the basis for access management at large scale.

4 System Model and Architecture

Centralized access control systems suffer from various problems such as: (a) the risk of privacy leakage, and (b) the risk of a single point of failure; (c) interoperability issues; (d) unreliability of the access control system, and (e) the presence of third parties.

An access control system can utilize the blockchain technology to address these problems. The decentralized nature of the blockchain resolves the problem of a single point of failure. Cryptographic methods ensure the reliability of the ledger. Consensus mechanisms ensure that the state of the ledger is valid, and it is the same for every participant. Smart contracts allow monitoring and enforcement of sophisticated access control decisions. Also, with automatic enforcement, they can address privacy issues.

This solution empowers both resource owners and subjects (typically access requesters). Details of each granted or revoked access permission can be queried from the ledger, including the policies that have been applied, the attribute values and the time of access request. In practice, under no circumstances, resource owners do not deny access to a resource by a rightful requester. On the other hand, the resource owners leverage provided audit trails, while being assured that no user had subverted the system.

To provide a solution to these problems, we used the Hyperledger Fabric blockchain. In the blockchain, there have to be at least two endorsing nodes belonging to different organizations. These nodes are responsible for executing SPs. Clients would be the systems that use this system, as depicted in Fig. 4.

The workflow of the users remains unaltered. The only difference is that the authorization requests are now mediated through one or more nodes representing the given system, as the evaluation of access control policies could be not trusted for the subject of the request, who instead requests invalid access to resources.

In order to protect the privacy of users' data, Hyperledger Fabric provides a private data feature to protect sensitive users' data. We have used this feature to represent the attributes that are required for access permissions based on the organizations' defined policies. The private data is hashed, and then it will be endorsed and ordered like other data. Finally, the chaincode writes hashed data on the ledgers of every peer. However, only organizations that require these private attributes to give access permission have access to them. Using Zero-knowledge proofs Zero knowledge proofs (ZKP)[18] is another alternative that can be applied for highly privacy-preserving case studies that require to protect all users' attributes from all access

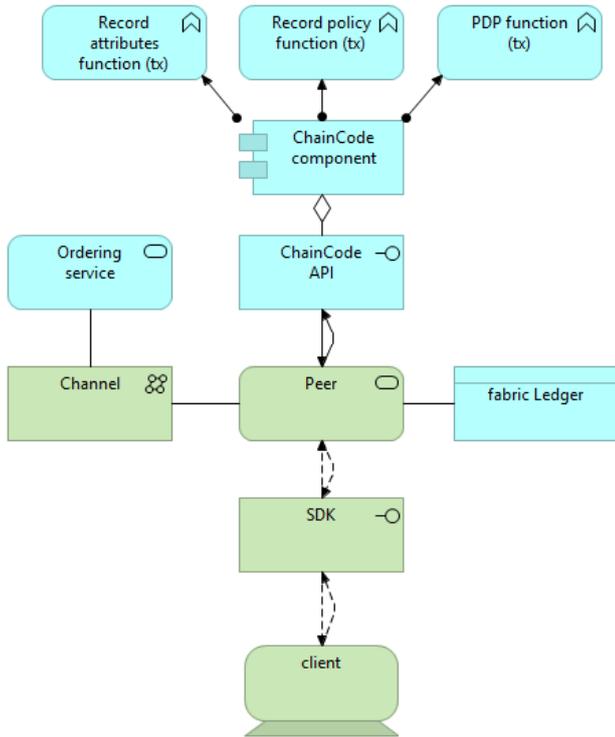


Fig. 4 High level system architecture using the Archimate modeling language [48].

providers and data owners. However, ZKP requires additional time and computational resources compared to our solution based on Hyperledger Fabric private data feature.

In our solution, as depicted in Figure 5, the blockchain acts as a mediator between the entity that requests access to a specific resource and the entity that manages that resource. The system includes two main components. The first component is an off-chain system that relies on permissioned blockchain to store its access control attributes on it and query access permissions from it. The second component is a permissioned blockchain that manages different access control components through smart contracts and stores the data on a tamper-proof ledger.

The three main smart contracts are PIP contract, PAP smart contract and the PDP contract. Subject (users) and resource (objects) attributes are stored in a JavaScript Object Notation (JSON) data format through the PIP contract. The PIP is also responsible for checking write conflicts and updating attributes. Policies are also recorded in the system as JSON data format, and PAP contract is responsible for managing policies and updating policies. The system could be implemented to work with multiple PAPrun by different organizations. Even if there is only one PAP, the trans-

parency offered by this solution distributes the trust and the responsibility of these access policies. PDP contract evaluates policies to make an access decision. Figure 6 shows the architecture of the implemented smart contracts.

After smart contracts evaluate SPs against their respective attributes, the PDP returns its decision to the PEP. This process allows a decoupling between users and the blockchain administration (as users do not need to have a node on the blockchain, which is desirable).

Our solution not only logs all access requests in a very secure way but also provides a framework to control all access controls concerning the participants in the network. Nodes from the private network can access the blockchain, check transactions' history, and audit the history of access requests and results. Automatic auditing techniques can be developed by analyzing the history of access request transactions. A fine-grained access control solution is provided that enforces access validation through blockchain-based service providers.

5 Case Study

A digital library is a collection of documents in an organized electronic form that allows users to access them online. A highly dynamic user population and the numerous collection of resources in digital libraries require a fine-grained and dynamic access control method such as ABAC [1]. It requires that access policies specified based on users' attributes and characteristics rather than users' roles in the system.

In this section, we have selected a case study for access control in digital libraries to illustrate and explain the application of our ABAC system.

For every subject, we store it with a subject ID (SID), and the set of its attributes and values, and the same for Objects attributes. Attribute ID is a required field to store attributes in the Hyperledger Fabric database and later retrieve the respective attribute for permission decision. Both Hyperledger Fabric supporting databases (CouchDB and Level DB) are key-value stores, and the ID field is used as keys.

S_nA is the set of attributes associated with the $subject_n$ and S_nID defined as a key for storing the correlated attributes. Similarly, O_nA is the set of attributes associated with the $object_n$ and O_nID defined as a key for storing the correlated attributes.

P_nSA and P_nOA are the sets of subjects and Objects determinative attributes in the policy of n . Same as attributes, Policy ID (P_nID) is a required field to store attributes. For every policy, we store the determinative attributes (P_nSA, P_nOA) along with the policy's rules.

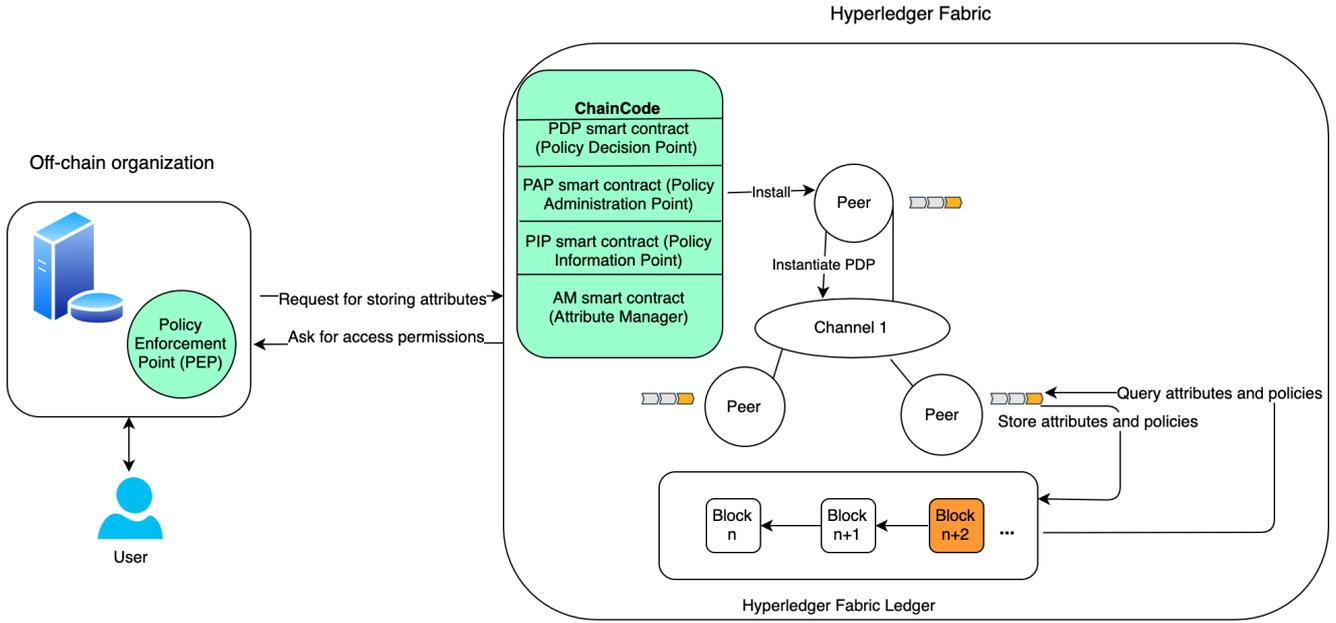


Fig. 5 Blockchain Access Control System Architecture.

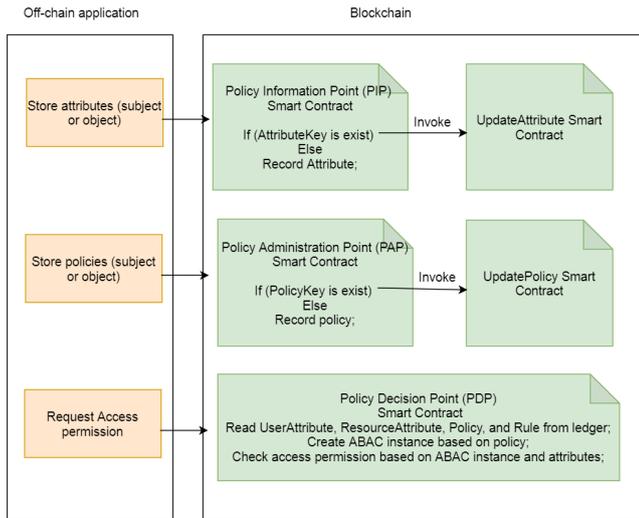


Fig. 6 Smart contracts architecture

$$S_nA = \{S_nID, \{\{S_nA_1, value\}, \dots, \{S_nA_n, value\}\}\}$$

$$O_nA = \{O_nID, \{\{O_nA_1, value\}, \dots, \{O_nA_n, value\}\}\}$$

Defining SA as the set of all subjects attributes and OA as the set of all objects attributes, we have:

$$SA = (S_1A \cup S_2A \cup \dots S_nA)$$

$$OA = (O_1A \cup O_2A \cup \dots O_nA)$$

$$P_nSA \subseteq SA$$

$$P_nOA \subseteq OA$$

$$P_nSA, P_nOA \in P$$

$$P_n = \{P_nID, P_nSA, P_nOA, rules\}$$

Resulting, we have the following attributes for the subject S_1A and object O_1A :

$$S_1A = \{“s001”, \{“status”, true\}, \{“expiration”, “2020-05-12”\}, \{“libraryGroup”, 12\}\}$$

$$O_1A = \{“r001”, \{“libraryGroup”, 12\}\}$$

The policy P_1 with the id “policy01” is as following:

$$P_1 = \{“policy01”, S_1A, O_1A, \{“status == true” \wedge “expiration” > “1Day” \wedge “user.libraryGroup” == “resource.libraryGroup”\}\}$$

5.1 JSON Data Format

In our implemented solution, the access control data (attributes and policies) is followed by the JSON format. Using JSON, as a widespread data format, can be used by a broad range of applications. An example of a sample policy, including subject (user) attributes, and object (resource) attributes, is illustrated in the following JSON code snippet. Based on the presented policy, the subject and the resource attributes, our subject has valid access permission to the resource, as the subject has valid Identifier (ID), active status, non-expired membership and the subject library group matches with the resource library group. If one of the subject’s attributes does not pass the policy rules, the access permission will be denied.

Example: Definition of a sample Access Control Policy, Subject and Resource followed by the JSON format

```
policy = {
  "policyID": "policy01",
  attributes: {
    "user": {
      "status": "Active",
      "expiration": "Date of expiration",
```

```

    "libraryGroup": "Group ID",
  },
  "resource": {
    "libraryGroup": "Group ID",
  },
  "rules": {
    "user.status": {
      "comparison_type": "boolean",
      "comparison": "boolAnd",
      "value": true
    },
    "user.expiration": {
      "comparison_type": "datetime",
      "comparison": "isMoreRecentThan",
      "value": "1DAY"
    },
    "user.libraryGroup": {
      "comparison_target": "libraryGroup",
      "comparison_type": "numeric",
      "comparison": "isStrictlyEqual",
      "field": "resource.libraryGroup"
    }
  }
}

subject = {
  subjectID: "s001",
  attributes: {
    "status": true,
    "expiration": "2020-05-12",
    "libraryGroup": 12
  }
}

resource = {
  resourceID: "r001",
  Attributes: {
    "libraryGroup": 12
  }
}

```

6 System evaluation

In this section, we evaluate the proposed system performance. We used Hyperledger Caliper to measure the performance of our system based on our own written benchmark and various configurations.

6.1 Environment configuration, performance parameters, and assumptions

We evaluate every component of our system against two different databases, Couchdb and Goleveldb and two orderer services, Raft and Kafka. We also present the results of the evaluation based on the Solo orderer to illustrate the effect of other parameters separated from the effect of the involved consensus method.

Raft is a crash fault-tolerant (CFT) ordering service based on the implementation of Raft protocol. Raft follows the “leader and follower” model. The leader makes decisions, and the followers follow the leader. The peer that represents the leader is changed frequently. Every follower has the chance to be a candidate to become the leader of the next round.

Similar to Raft, Kafka is a CFT ordering service, which follows the “leader and follower” model as well. However, Kafka uses ZooKeeper⁵ to manage clusters. Zookeeper keeps track of the status of the Kafka cluster nodes and partitions.

Solo is a single ordering node, and it is not fault-tolerant. It is meant to be used for testing purposes.

We used the Google Cloud Platform to run a Virtual Machine (VM) instance and test our application and collect performance analysis data. The machine type is n2-highcpu-8, which includes eight virtual CPUs and 8 GB of memory. All the tests are run on the same virtual machine, as Caliper emulates workload distribution between several clients.

The default number of blockchain clients is 10 (each client emulated by a different NodeJS⁶ process), the default number of transactions is 5,000, and the default transaction type is policy decision transaction, which queries the related data from ledger based on access request and determines the result of the access request. The default database for Raft and Kafka is GoLevelDB. The default number of organizations is two, and the default number of peers is one.

6.2 Performance evaluation results

Figure 7 shows the average latency (in seconds) for three different transactions, Record attributes, PDP, and query data from the ledger based on Kafka orderer. In every round of the test, we configured the test with a different number of transactions. Although the average latency increases with the number of transactions, the increase is not sharp, and it increases very slowly. As the graph illustrates, the system can process 10,000 access decisions with an average latency of 0.54 seconds.

Figure 8 shows the average latency (in seconds) for the same three different transactions, based on the Raft orderer. The test result is based on a different number of transactions. Similarly, the average latency increases with the increase in the number of transactions. The test run has failed for the Raft orderer with 10,000 transactions due to VM Memory limitation. The resource consumption result indicates that Raft memory

⁵ <https://zookeeper.apache.org/>

⁶ <https://nodejs.org/en/>

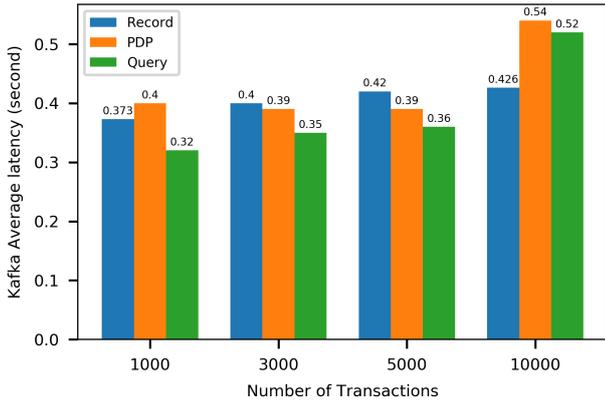


Fig. 7 Average latency of Kafka as a function of the type of transactions.

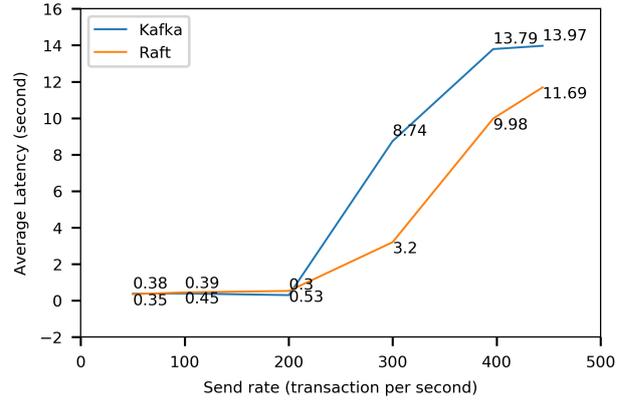


Fig. 9 Average latency of Raft and Kafka based on different transactions.

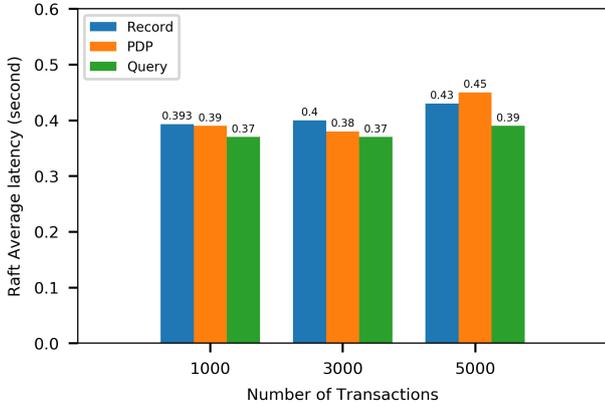


Fig. 8 Average latency of Raft under different transactions.

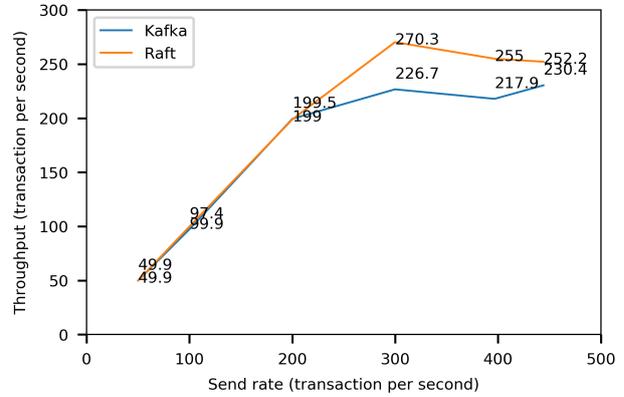


Fig. 10 Throughput of Raft and Kafka under different transactions.

consumption increases 4.33 times, in comparison with Kafka. This fact explains why the test failed in the middle of executing 10,000 transactions with the Raft orderer.

For both Raft and Kafka orderers, increasing the number of transactions increases the average latency for the record attributes transaction. Policy decision transaction has the minimum average latency for both Kafka and Raft, based on 3,000 transactions. For Record attribute and query data transactions, the average latency in Raft is slightly higher than Kafka. For the policy decision transaction, the average latency in Raft for 1,000 and 3,000 transactions is slightly lower than Kafka, but for 5,000 transactions, the result is the opposite.

Figures 9 and 10 show the average latency and throughput for the policy decision transaction for Raft and Kafka based on different send rates. The number of transactions for these two tests is 5,000. The dendrite of 200 transactions per second (tps) is an optimal point for Kafka as it exhibits the lowest average latency, and

the average latency increases sharply after the send rate of 200 tps.

The maximum throughput for both Raft and Kafka orderers is at the send rate point of 300 tps; afterwards, the throughput drops for both of them. Overall, in terms of throughput and average latency, Raft performed better than Kafka when the throughput passed 200 tps as a turning point for Kafka.

Figure 11 shows the effect of increasing the number of organizations and peers and the comparison of two different databases, GoLevelDB and CouchDB. Increasing the number of organizations and peers increases the average latency for all three transactions. For the CouchDB database with three organizations and two peers, the average latency increases sharply to 43.81 seconds for the policy decision transaction, which is 17.73 times more than GolevelDB and 64.42 times more than the same database, with two organizations and one peer. It shows that CouchDB performs inadequately in comparison with GoLevelDB.

Table 2 Resource consumption for Raft and Kafka.

Orderer	Name	Memory(max)	Memory(avg)	CPU(max)	CPU(avg)	Traffic In	Traffic Out	Disc Read	Disc Write
Raft	dev-peer0.org1	74.4MB	71.1MB	32.25%	28.21%	13.8MB	5.3MB	0B	0B
	dev-peer0.org2	73.3MB	69.7MB	33.23%	28.28%	13.8MB	5.3MB	0B	0B
	peer0.org1	379.3MB	369.4MB	67.21%	56.32%	31.1MB	23.7MB	0B	21.8MB
	peer0.org2	284.0MB	274.1MB	70.78%	55.66%	31.1MB	23.6MB	4.0KM	21.8MB
	orderer1	554.1MB	535.4MB	26.11%	15.41%	22.4MB	59.1MB	0B	37.2MB
	orderer2	525.3MB	506.5MB	18.78%	11.88%	27.6MB	28.7MB	0B	37.0MB
	orderer0	513.3MB	494.7MB	19.40%	11.63%	27.5MB	10.6MB	0B	37.2MB
Kafka	dev-peer0.org1	73.5MB	72.8MB	17.87%	15.26%	7.5MB	2.5MB	0B	0B
	dev-peer0.org2	64.5MB	62.8MB	18.73%	15.69%	7.5MB	2.5MB	0B	0B
	peer0.org1	295.9MB	286.2MB	52.08%	49.15%	27.1MB	17.0MB	368.0KB	21.2MB
	peer0.org2	294.1MB	282.7MB	51.54%	48.38%	27.1MB	17.1MB	152.0KB	21.2MB
	orderer0	121.1MB	113.1MB	25.80%	23.30%	29.6MB	11.1MB	4.0KB	18.4MB
	orderer1	124.1MB	115.2MB	21.87%	20.25%	29.7MB	46.5MB	276.0KB	18.4MB
	orderer2	121.1MB	113.1MB	25.80%	23.30%	29.6MB	11.1MB	4.0KB	18.4MB

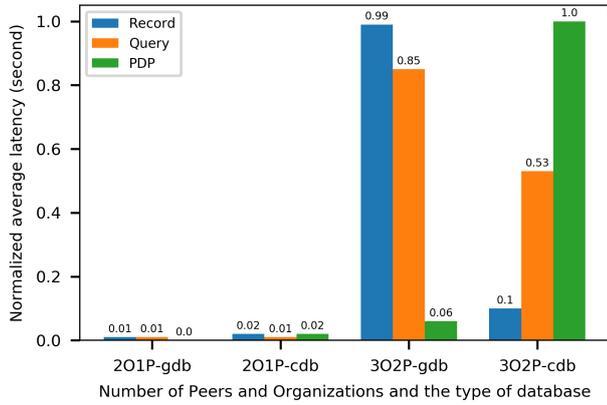
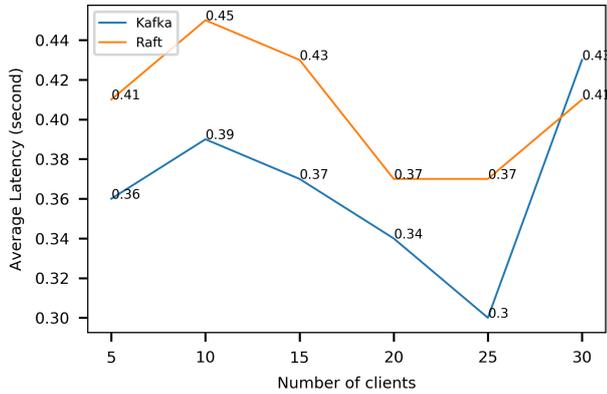
**Fig. 11** Throughput of Raft and Kafka under different transactions. Legend: $xOnP = x$ Organizations and n Peers; gdb = GoLevelDB; cdb = CouchDB.**Fig. 12** Average latency based of Raft and Kafka with different number of clients.

Figure 12 shows the average latency for both Raft and Kafka based on the different number of clients. This test is run based on 5,000 transactions and the policy decision transaction. For Kafka, 25 number of clients is like an optimal point that has the lowest average latency (0.3 seconds). However, 25 is an optimal point for the system with current resources. For Raft, as the

graph shows, there are two points for the minimum average latency, corresponding to 20 and 25 number of clients. In general, Kafka performs better under 25 clients, but after 25 clients, it shows a sharp increase in the average latency. Although it shows that the system is not scalable after 25 clients, it significantly depends on the computation power and limitations of the VM instance. We have repeated the test with a more powerful VM instance, and average latency was 0.36 second for Raft and 0.31 second for Kafka with 60 clients.

Table 2 presents the resource consumption for policy decision transactions based on 5000 transactions for Kafka and Raft. As the presented numbers in the table demonstrate, Raft consumes 4.33 times more memory on average in comparison with Kafka. It clarifies that our early test with 10,000 transactions with Raft ordered failed because the VM ran out of memory.

7 Conclusion

Dependable accountability mechanisms are essential for audits. In this paper, we discussed how permissioned blockchains could be helpful as a trustable backend in access control systems, thus providing a solid basis for audits. We proposed a distributed ABAC system based on Hyperledger Fabric with a focus on auditability and scalability. We validated our solution through a decentralized access control management application in digital libraries. First, we presented a comprehensive review of studies focusing on blockchain-based access control studies. Then we presented the system architecture and implementation details, where the PDP, PAP, and AM components have been implemented using smart contracts on-chain, and the PEP was implemented off-chain - based on the blockchain clients' requirements.

The experimental evaluation of our solution considered various parameters based on the Hyperledger Caliper framework in terms of system performance. The evaluation results indicate that our system can effec-

tively handle 10,000 access request transactions with an average latency of 0.54 seconds.

Future work is in progress in two directions: first, building a robust framework and platform-independent solution towards distributed access control; second, integrating user authentication to our authorization solution.

Acknowledgement

This research is supported by the Linux Foundation, in the context of the Hyperledger Fabric Based Access Control Project.

References

- Adam, N.R., Atluri, V., Bertino, E., Ferrari, E.: A content-based authorization model for digital libraries. *IEEE Transactions on knowledge and data engineering* **14**(2), 296–315 (2002)
- Alansari, S., Paci, F., Sassone, V.: A distributed access control system for cloud federations. In: *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*, pp. 2131–2136. IEEE (2017)
- Anderson, A., Parducci, B., Adams, C.: OASIS eXtensible Access Control Markup Language (XACML). Online, ... (2006). URL http://research.sun.com/projects/xacml/XMLCOP_{_}060620{_}slides.pdf
- Androulaki, E., Barger, A., Bortnikov, V., Muralidharan, S., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Murthy, C., Ferris, C., Laventman, G., Manevich, Y., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., Cocco, S.W., Yellick, J.: Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. *Proceedings of the 13th EuroSys Conference, EuroSys 2018 2018-Janua* (2018). DOI 10.1145/3190508.3190538
- Azaria, A., Ekblaw, A., Vieira, T., Lippman, A.: MedRec: Using blockchain for medical data access and permission management. *Proceedings - 2016 2nd International Conference on Open and Big Data, OBD 2016* pp. 25–30 (2016). DOI 10.1109/OBD.2016.11
- Belchior, R., Correia, M., Vasconcelos, A.: JusticeChain: Using Blockchain To Protect Justice Logs. In: *CoopIS 2019: 27th International Conference on Cooperative Information Systems* (2019)
- Belchior, R., Vasconcelos, A., Correia, M.: Towards Secure, Decentralized, and Automatic Audits with Blockchain. In: *European Conference on Information Systems* (2020)
- Bell, E.D., La Padula, J.L.: Secure computer system: Unified exposition and multics interpretation (1976)
- Bertino, E., Weigand, H.: An approach to authorization modeling in object-oriented database systems. *Data & Knowledge Engineering* **12**(1), 1 – 29 (1994)
- Biba, K.: Integrity considerations for secure computer systems. Tech. rep., Bedford, MA: Mitre Corporation (1977)
- Dagher, G.G., Mohler, J., Milojkovic, M., Marella, P.B.: Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. *Sustainable Cities and Society* **39**(February), 283–297 (2018). DOI 10.1016/j.scs.2018.02.014. URL <https://doi.org/10.1016/j.scs.2018.02.014>
- Di Francesco Maesa, D., Mori, P., Ricci, L.: A blockchain based approach for the definition of auditable Access Control systems. *Computers and Security* **84**, 93–119 (2019). DOI 10.1016/j.cose.2019.03.016. URL <https://doi.org/10.1016/j.cose.2019.03.016>
- Ding, S., Cao, J., Li, C., Fan, K., Li, H.: A Novel Attribute-Based Access Control Scheme Using Blockchain for IoT. *IEEE Access* **7**, 38431–38441 (2019). DOI 10.1109/ACCESS.2019.2905846
- Dukkipati, C., Zhang, Y., Cheng, L.C.: Decentralized, blockchain based access control framework for the heterogeneous internet of things. In: *Proceedings of the Third ACM Workshop on Attribute-Based Access Control*, pp. 61–69. ACM (2018)
- Es-Samaali, H., Outchakoucht, A., Leroy, J.P.: A blockchain-based access control for big data. *International Journal of Computer Networks and Communications Security* **5**(7), 137 (2017)
- Ferdous, M.S., Margheri, A., Paci, F., Yang, M., Sassone, V.: Decentralised runtime monitoring for access control systems in cloud federations. In: *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*, pp. 2632–2633. IEEE (2017)
- Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn, D., Chandramouli, R.: A Proposed Standard for Role-Based Access Control. *ACM Transactions on Information and System Security* **4**(3) (2001)
- Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology* **7**, 1–32 (1994)
- Guo, H., Li, W., Nejad, M., Shen, C.C.: Access control for electronic health records with hybrid blockchain-edge architecture. *arXiv preprint arXiv:1906.01188* (2019)
- Guo, H., Meamari, E., Shen, C.C.: Multi-authority attribute-based access control with smart contract. In: *Proceedings of the 2019 International Conference on Blockchain Technology*, pp. 6–11. ACM (2019)
- Hu, S., Hou, L., Chen, G., Weng, J., Li, J.: Reputation-based distributed knowledge sharing system in blockchain. In: *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp. 476–481. ACM (2018)
- Hu, V.C., Ferraiolo, D., Kuhn, R., Friedman, A.R., Lang, A.J., Cogdell, M.M., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K., et al.: Guide to attribute based access control (abac) definition and considerations (draft). NIST special publication **800**(162) (2013)
- Hu, V.C., Kuhn, D.R., Ferraiolo, D.F.: Access control for emerging distributed systems. *Computer* **51**(10), 100–103 (2018). DOI 10.1109/MC.2018.3971347
- Jemel, M., Serhrouchni, A.: Decentralized access control mechanism with temporal dimension based on blockchain. In: *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*, pp. 177–182. IEEE (2017)
- Khan, M.A., Salah, K.: IoT security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems* **82**, 395–411 (2018). DOI 10.1016/j.future.2017.11.022. URL <https://doi.org/10.1016/j.future.2017.11.022>
- Kuo, T.T., Kim, H.E., Ohno-Machado, L.: Blockchain distributed ledger technologies for biomedical and health

- care applications. *Journal of the American Medical Informatics Association* **24**(6), 1211–1220 (2017)
27. Lee, Y., Lee, K.M.: Blockchain-based rbac for user authentication with anonymity. In: *Proceedings of the Conference on Research in Adaptive and Convergent Systems*, pp. 289–294. ACM (2019)
 28. López-Pintado, O., García-Bañuelos, L., Dumas, M., Weber, I.: Caterpillar: A blockchain-based business process management system. In: *Proceedings of the BPM Demo Track and BPM Dissertation Award co-located with 15th International Conference on Business Process Modeling (BPM 2017)*, Barcelona, Spain (2017)
 29. Lyu, Q., Qi, Y., Zhang, X., Liu, H., Wang, Q., Zheng, N.: Sbac: A secure blockchain-based access control framework for information-centric networking. *Journal of Network and Computer Applications* **149**, 102444 (2020)
 30. Ma, M., Shi, G., Li, F.: Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the iot scenario. *IEEE Access* **7**, 34045–34059 (2019)
 31. Maesa, D.D.F., Mori, P., Ricci, L.: Blockchain based access control. In: *IFIP international conference on distributed applications and interoperable systems*, pp. 206–220. Springer (2017)
 32. Maryline, L., Nesrine, K., Christian, L.: A Blockchain based Access Control Scheme. In: *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications*, pp. 168–176 (2018)
 33. Novo, O.: Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT. *IEEE Internet of Things Journal* **5**(2), 1184–1195 (2018). DOI 10.1109/JIOT.2018.2812239
 34. Novo, O.: Blockchain meets iot: An architecture for scalable access management in iot. *IEEE Internet of Things Journal* **5**(2), 1184–1195 (2018)
 35. Ouaddah, A., Abou Elkalam, A., Ait Ouahman, A.: Fairaccess: a new blockchain-based access control framework for the internet of things. *Security and Communication Networks* **9**(18), 5943–5964 (2016)
 36. Outchakoucht, A., Hamza, E., Leroy, J.P.: Dynamic access control policy based on blockchain and machine learning for the internet of things. *Int. J. Adv. Comput. Sci. Appl* **8**(7), 417–424 (2017)
 37. Paillisse, J., Subira, J., Lopez, A., Rodriguez-Natal, A., Ermagan, V., Maino, F., Cabellos, A.: Distributed access control with blockchain. *arXiv preprint arXiv:1901.03568* (2019)
 38. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: *Annual International Cryptology Conference*, pp. 129–140. Springer (1991)
 39. Pinno, O.J.A., Gregio, A.R.A., De Bona, L.C.: ControlChain: Blockchain as a Central Enabler for Access Control Authorizations in the IoT. *2017 IEEE Global Communications Conference, GLOBECOM 2017 - Proceedings 2018-Janua*, 1–6 (2018). DOI 10.1109/GLOCOM.2017.8254521
 40. Pinno, O.J.A., Grégio, A.R.A., De Bona, L.C.: Controlchain: A new stage on the iot access control authorization. *Concurrency and Computation: Practice and Experience* p. e5238 (2019)
 41. Pourheidari, V., Rouhani, S., Deters, R.: A Case Study of Execution of Untrusted Business Process on Permissioned Blockchain. *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) (September)*, 1129–1136 (2018). DOI 10.1109/Cybermatics
 42. Rajput, A.R., Li, Q., Ahvanooy, M.T., Masood, I.: Eacms: emergency access control management system for personal health record based on blockchain. *IEEE Access* **7**, 84304–84317 (2019)
 43. Rouhani, S., Butterworth, L., Simmons, A.D., Humphery, D.G., Deters, R.: MediChain™: A Secure Decentralized Medical Data Asset Management System. *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) (September)*, 1129–1136 (2018). DOI 10.1109/Cybermatics
 44. Rouhani, S., Deters, R.: Blockchain based access control systems: State of the art and challenges. In: *IEEE/WIC/ACM International Conference on Web Intelligence, WI '19*, pp. 423–428. ACM, New York, NY, USA (2019). DOI 10.1145/3350546.3352561. URL <http://doi.acm.org/10.1145/3350546.3352561>
 45. Rouhani, S., Deters, R.: Security, performance, and applications of smart contracts: A systematic survey. *IEEE Access* **7**, 50759–50779 (2019). DOI 10.1109/ACCESS.2019.2911031
 46. Rouhani, S., Pourheidari, V., Deters, R.: Physical Access Control Management System Based on Permissioned Blockchain. In: *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) (2019)*
 47. Sandhu, R.S., Samarati, P.: Access control: principle and practice. *IEEE Communications* **32**(9), 40–48 (1994)
 48. TO Group: ArchiMate®3.0 Specification. Van Haren Publishing (2016)
 49. Wang, S., Zhang, Y., Zhang, Y.: A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access* **6**, 38437–38450 (2018). DOI 10.1109/ACCESS.2018.2851611
 50. Wang, S., Zhang, Y., Zhang, Y.: A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access* **6**, 38437–38450 (2018)
 51. Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A., Mendling, J.: Untrusted business process monitoring and execution using blockchain. In: *International Conference on Business Process Management*, pp. 329–347. Springer (2016)
 52. Xia, Q., Sifah, E.B., Asamoah, K.O., Gao, J., Du, X., Guizani, M.: Medshare: Trust-less medical data sharing among cloud service providers via blockchain. *IEEE Access* **5**, 14757–14767 (2017)
 53. Xu, R., Chen, Y., Blasch, E., Chen, G.: Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness. *Optical Engineering* **58**(4), 041609 (2019)
 54. Yuan, E., Tong, J.: Attributed based access control (abac) for web services. In: *IEEE International Conference on Web Services (ICWS'05)*. IEEE (2005)
 55. Zhang, X., Poslad, S.: Blockchain support for flexible queries with granular access control to electronic medical records (emr). In: *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6. IEEE (2018)
 56. Zhang, X., Poslad, S.: Blockchain Support for Flexible Queries with Granular Access Control to Electronic Medical Records (EMR). *IEEE International Conference on*

-
- Communications **2018-May** (2018). DOI 10.1109/ICC.2018.8422883
57. Zhang, Y., Kasahara, S., Shen, Y., Jiang, X., Jianxiong-Wan: Smart Contract-Based Access Control for the Internet of Things. *IEEE Internet of Things Journal* **6**(2), 1594–1605 (2019)
 58. Zhu, Y., Qin, Y., Gan, G., Shuai, Y., Chu, W.C.C.: TBAC: Transaction-Based Access Control on Blockchain for Resource Sharing with Cryptographically Decentralized Authorization. *Proceedings - International Computer Software and Applications Conference* **1**, 535–544 (2018). DOI 10.1109/COMPSAC.2018.00083
 59. Zhu, Y., Qin, Y., Gan, G., Shuai, Y., Chu, W.C.C.: Tbac: transaction-based access control on blockchain for resource sharing with cryptographically decentralized authorization. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 1, pp. 535–544. IEEE (2018)
 60. Zyskind, G., Nathan, O., Pentland, A.S.: Decentralizing Privacy: Using Blockchain to Protect Personal Data. In: *IEEE Security and Privacy Workshops*, pp. 180–184 (2015)