Utilizing Policy Machine for Attribute-Based Access Control in Permissioned Blockchain

Sherifdeen Lawal
Univ. of Texas at San Antonio
Texas, USA
sherifdeenlawal@gmail.com

Ram Krishnan
Univ. of Texas at San Antonio
Texas, USA
ram.krishnan@utsa.edu

2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS) | 978-1-6654-3156-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/COINS51742.2021.9524248

***Abstract*—In recent times, as the number of blockchain use cases continues to grow, methods and technologies utilized by fraudsters continue to become sophisticated. The most notable form of cyber-attack utilizes a security breach in the internal security of blockchain systems, leading to illegal access to application services. A complex system like the blockchain network requires a dynamic, flexible, and scalable access control mechanism. There are numerous research efforts to leverage smart contracts in implementing access control based on blockchain. However, most of these contributions are either focused on blockchain-based access control for an off-chain resource. Other effects implement blockchain-based access control for a specific domain. We present a generic system architecture with a deployment on Hyperledger Fabric using the NIST Next Generation Access Control (NGAC) for deciding access to the protected resources on the blockchain.**

***Keywords*—Attribute-Based Access Control, Blockchain, Hyperledger Fabric, Policy Machine, Smart Contract**

I. Introduction

Permissionless (public) and pemissioned (private) blockchain are classifications based on the identity of the participants on the network. Permissionless blockchains allow any participant to join anonymously. Well-known examples of permissionless blockchains are blockchains such as Bitcoin, Ethereum, Litecoin, Dash, and Monero. The permissioned blockchains require the participant to obtain permission from a centralized or federated authority to join the network with a known identity. Popular blockchains such as Quorum, Hyperledger Fabric, Corda, and MultiChain are in this category. The permissioned blockchain thrives in enterprise use cases where the identity of the participants is a hard requirement, for instance, a financial transaction that must follow the Know-Your-Customer (KYC) and Anti-Money Laundering (AML) regulations. We focus on the Hyperledger Fabric blockchain network for our study in this paper. Hyperledger Fabric is an open-source, modular, and configurable architecture for enterprise-grade permissioned distributed ledger technology (DLT) platform.

A blockchain network exemplifies a particular set of challenges for distributed system access control. It requires a different set of concepts and considerations from traditional systems. A critical requirement is that distributed applications on other coordinated systems have permission to access the data for processing and deal with the access to the distributed processes and data from their local users. A solution for the described problem is well studied. That solution is an Attribute-Based Access Control (ABAC) for the enforcement access policy on the blockchain network. The ABAC model is suitable for blockchain network access control. It provides flexibility through attributes and enables the assignment of privilege in a distributed system that requires federated and autonomous control. In the Hyperledger Fabric, an application can interact with a blockchain network by submitting transactions to a ledger or querying ledger content. An ABAC approach utilized to control the interaction of an application with a blockchain network is a logic-based policy.

However, an ABAC policy expressed as the logic-based formula is (a boolean satisfiability problem) NP-complete in evaluating, for instance, a user attribute to access a particular resource. More succinctly, logic-based ABAC models such as XACML have been shown empirically to lack scalability [8]. For an alternative approach, Mell et al. demonstrated that the NIST Next Generation Access Control (NGAC), an enumerated-based policy ABAC model is scalable [2].

We have proposed a modularized ABAC architecture of the Policy Machine as an on-chain mechanism to control access to on-chain resources (assets). The Policy Machine is the basis for the NIST Next Generation Access Control (NGAC) [17]. As a proof-of-concept, we implemented the proposed architecture on the Hyperledger Fabric network. In our implementation, we log access policy information to the blockchain ledger through a set of smart contracts (chaincode). There is a separate blockchain ledger for the protected resource. A low-level Hyperledger fabric API facilitates the transfer of request and response between the Smart contracts deployed for access policy information and protected resource.

In summary, the contributions of this paper include:

1) the first-ever implementation of the NIST NGAC system in a blockchain network.
2) an illustration of an instance of the Policy Machine for controlling access to assets in multiple blockchain ledgers.

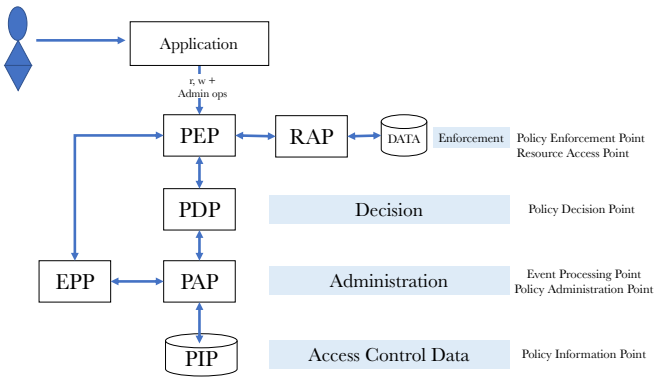

Fig. 1. NGAC Standard Functional Architecture (Adapted from [16])

The remainder of this paper is structured as follows. Section II discusses related work. Policy Machine and Hyperledger Fabric background is presented in Section III. Section IV provides a system architecture and implementation of Policy Machine on Hyperledger Fabric blockchain network. Section V presents a use case and access request scenarios. The performance evaluation is discussed in section VI, and section VII gives our conclusions.

II. Related Work

In this section, we overview previous research contributions on exploiting blockchain for an ABAC implementation. The application of blockchain as an Attribute-Based Access Control system is studied in different domains. The implementation of blockchain-based ABAC in IoT systems is discussed by Pinno et al. [19], Ding et al. [18], and Dukkipati et al. [7]. Zhang and Posland in [20] propose a blockchain-oriented authorization scheme for Electronic Medical Records (EMRs). The granularity of their authorization approach support queries of blocks and attribute values. Also, their method lowers the computational overhead by eliminating the use of the Public Key Infrastructure (PKI) for authorization, encryption, and decryption.

Another attempt to control access to Electronic Health Record (EHR) proposes an architecture that uses both blockchain and edge node [21]. The blockchain serves as the controller that manages the identity and access control policies specified in the Abbreviated Language For Authorization (ALFA). Besides, the edge node is off-chain storage for the EHR data and uses specified policies to enforce attribute-based access control on EHR data in combination with blockchain-based access control logs.

Maesa et al. [1] introduced an access control service utilizing the Ethereum blockchain platform. The blockchain stores smart contracts as access control policies specified using eXtensible Access Control Markup Language (XACML) [23] and process access decision making. Gou et al. [22] propose a multi-authority attribute-based access control scheme by using Ethereum's smart contracts. The Ethereum smart contracts provide the specification for the interactions between data owner, data user, and multiple attribute authorities. A data user presents its attributes to different attribute authorities and receives attributes tokens from respective attribute authorities only if validation of attributes is successful. The use of a distributed Attribute-Based Access Control system based on Hyperledger Fabric blockchain to provide trusted auditing of access attempts was proposed by Rouhani et al. [24].

There are few implementations of attribute-based access control on a blockchain for a domain-independent scenario [1], [22], [24]. While [1], [24] specify policies using the XACML, we utilize the Policy Machine, a different open-source attribute-based access control framework developed by NIST. Rouhani et al. [24], the only generic implementation of attribute-based access control on Hyperledger Fabric blockchain network, utilizes ABAC components as smart contracts to control access to an off-chain system. In contrast, our attribute-based access control implementation specifies policies using smart contracts for access control to the on-chain data. Also, an all-purpose blockchain-oriented attribute-based access control implemented by Maesa et al. [1] and Guo et al. [22] utilizes the Ethereum blockchain platform. The Ethereum based blockchain implementation is costly because of the fee paid for every operation. All public blockchain charges for transactions, but there are no fees in the permissioned blockchain.

III. Background

A. Policy Machine

Policy Machine (PM) is a flexible access control framework that uses a standardized and generic set of relations and functions for access control mechanisms. It aims at providing a general and unified framework to support access control services in different environments and support multiple access control models simultaneously. In general, PM comprises policy elements (i.e., sets of users, objects, user attributes, object attributes, and policy classes), operations, processes, and access rights. Four types of relations the Policy Machine support are assignment, association, prohibition, and obligation. Policy specification for the Policy Machine has a Directed Acyclic Graph (DAG) representation. The policy elements are the nodes. Assignment relations are unlabeled edges, association and prohibition are edges labeled with a set of access rights. A group of one or more PM servers, PM client, PM database, and resource server is the PM standard architectural component. A four-layer representation of these components is shown in Figure 1. The PM server includes the policy decision point (PDP), the policy administration point (PAP), and the event processing point (EPP). The policy enforcement point (PEP) and the application that provides an interface for users to request access to protected resources makes the PM client. The policy information point (PIP) stores an abstract representation of all the information about the access control data (policy elements) and relations as the PM database. The use of Policy Machine to offer access control service for blockchain distributed applications in the

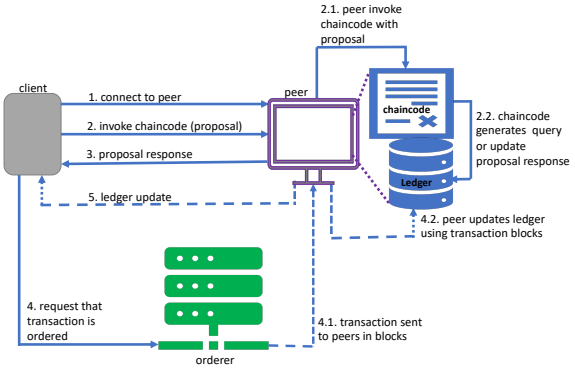

Fig. 2. The transaction flow for query and update proposal to the ledger

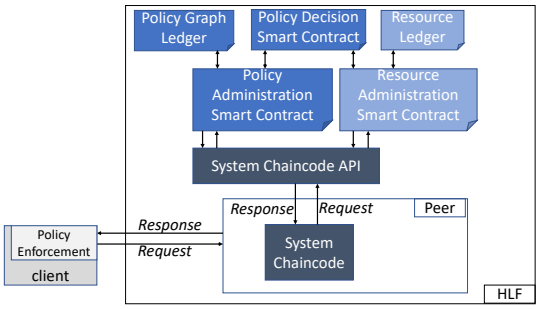

Fig. 3. Blockchain Access Control System Architecture.

Hyperledger Fabric network is of interest in this paper. Next, we describe its implementation in the section that follows.

B. Hyperledger Fabric

Hyperledger Fabric is a permissioned blockchain infrastructure that executes smart contracts written in general-purpose programming languages (e.g., Go, Java, Node.js). It utilizes well-known and proven technologies, with an architecture of pluggable modules for various functions. The core infrastructural components of the Hyperledger Fabric blockchain network comprise the distributed ledger, peer nodes (or simply nodes), chaincode, channel, and Membership Service Provider (MSP). Next, we introduce these components.

The Hyperledger Fabric ledger stores asset. An asset (or business object) is the digital representation for anything of value stored on peer nodes and is transferable between blockchain network participants. A ledger has the world state (or simply state) and the blockchain. The former is a database expressed as key-value pairs that hold the current value of an asset. The blockchain has a structure of a sequential log of interlinked blocks, where each block contains a sequence of transactions, each transaction representing an update to the world state.

There are three major types of nodes in the Hyperledger Fabric network. A client (client node) acts on behalf of the end-user for creating and submitting a request to the network. The Hyperledger Fabric Software Development Kits provides the platform to develop client nodes. Peer nodes are the fundamental element of the Hyperledger Fabric blockchain network as they host the ledgers that store the asset. Any node that renders the service of block sequencing and transaction sequencing within blocks is called the ordering node. An ordering node is critical to the instantiation of the network. At the launch of an ordering node, it produces the first block called the genesis block. The genesis block contains the network configuration properties and policies specified at the initialization of the orderer node.

A chaincode is the smart contract implementation in Hyperledger Fabric, a code that accesses the ledger and provides instructions for the asset query and modification. An instance of the blockchain network is called a channel. The channel serves as a mechanism for a set of components to communicate and transact privately. Metadata in the genesis block preserves the configuration of a channel. The Membership Service Provider (MSP) refers to an abstract component of the system that provides credentials to clients and peers to enable participation in the Hyperledger Fabric network. Clients use these credentials to authenticate their transactions, and peers use their credentials to validate a transaction.

After installing a chaincode dedicated for a channel on a peer node, end users can invoke the chaincode (smart contract) to read and update the ledger using a conforming application or CLI on the client node. A ledger query is a straightforward transaction. A peer can respond to a client application query instantly since the peer's local copy of the ledger has the required information to satisfy the query. Figure 2 shows a query request in three steps - the client application connects to a peer, invokes a chaincode with a query request, and the chaincode returns a query response. However, ledger update is a more complex interaction between the client application, peers, and orderer. In addition to three steps in the query transaction request, require two extra steps in an update transaction request. A single peer node can not respond to a ledger update transaction request. It is a consensus process among all the peers that have the ledger's copy. The designated chaincode on a peer node returns a simulated (i.e., results not effected on the ledger) results as step three of a ledger update transaction request.

IV. System Architecture and Implementation

In the previous section, we provide an overview of the Policy Machine. Figure 3 represents the functional architectural implementation of Policy Machine in the Hyperledger Fabric blockchain network. We implement the Policy Information Point (PIP) for storing the access control state and the database for the protected resources as Policy Information and Resource Ledgers, respectively. The Policy Administration Smart Contract represents the Policy Administration Point (PAP). It mediates access and enables modification to the Policy Information Ledger. Similarly, we implement the

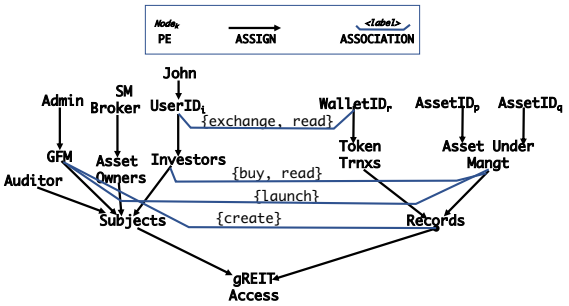

Fig. 4. Policy Machine Authorization Graph for global REIT

Scenario 1

```
[redacted]$ source John-login.sh
[redacted]$ peer chaincode query -C pmchannel -n
ttcc  -c '{"Args":["readAccount", "WalletID"]}'
{"ID":"WalletID","GREIT":"500","ERC20":"2000","accountType":"invest"}
[redacted]$ peer chaincode invoke
"${TARGET_TLS_OPTIONS[@]}" -C pmchannel -n ttcc  -c '{"Args":["buyGRET",
"WalletID", "KJKPlaza", "150"]}'
2021-05-08 13:13:24.756 CDT [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001
Chaincode invoke successful. result: status:200 payload:"650"
```

Scenario 2

```
[redacted]$ source SMBroker-login.sh
[redacted]$ peer chaincode invoke
"${TARGET_TLS_OPTIONS[@]}" -C pmchannel -n pgacc  -c
'{"Args":["launchTokenizedAsset", "UpTownHotel", "AUM", "gREITAccess"]}'
Error: endorsement failure during invoke. response: status:500 message:"SMBroker
is unauthorized to launch token asset: UpTownHotel"
```

Fig. 5. Policy Decision Smart contract response to access request scenarios

(RAP) as Resource Access Smart Contract that intercepts application (resource) user's request to the Resource Ledger. Our Policy Decision Smart Contract (see figure 3) is the logic that represents the Policy Decision Point (PDP) component of the Policy Machine. It makes access decisions on access requests forwarded from both Policy Administration Smart Contract and Resource Access Smart Contract response.

This architecture aims at implementing the Policy Machine for client nodes that are native to the blockchain. We need not implement a smart contract for the Policy Enforcement Point (PEP), and the Event Processing Point (EPP) component of the Policy Machine is outside the scope of this work.

Some components of the Policy Machine we implement as Smart contracts are in the go programming language. We leverage the invokeChaincode Application Programming Interface (API) to enable the request and response between Smart contracts (chaincode). For instance, assuming the installation of the smart contracts are on the same channel. The Resource Access Smart Contract conveys the access decision for an application users' request. It locally calls the invoke function of the Policy Decision Smart contract that returns access decision response. This inter-chaincode function call does not require a new transaction message. The calling chaincode uses the same transaction context as its caller. Note that invokeChaincode API allows the cases where the calling and called chaincode are on the same or different channel. However, for our implementation, only when an invocation of a chaincode by another on the same channel is allowed. Recall that the Policy Information Ledger stores an abstract representation of protected resources in the Resource Ledger. For consistency on these two ledgers, the Policy Decision Smart contract is allowed both read and write access on the two ledges. If the Policy Decision Smart Contract is not on the same channel with the two ledges, any (delete/create) modification request to these ledges will not affect.

V. Use Case And Scenarios

A case study for this work is a blockchain-based global Real Estate Investment Trust (REIT). The concept of global REIT is a portfolio diversifier designed to deliver high returns and income through investments in real estate investment trust (REIT) and real estate companies worldwide. It offers investors exposure to global real estate markets without the necessity of acquiring an entire property and shift the management and compliance obligations to the fund management. Apart from offering high total investment return through a combination of capital appreciation and current income like traditional REITS, blockchain-based global Real Estate Investment Trust (REIT) also provide (a) real estate investment using cryptocurrency (b) stable dividends for crypto investors (c) crypto domain for real estate asset holders to exit into a liquid market

A global REIT acquires assets around the world. Acquired assets, also called Asset Under Management (AUM), with a Net Asset Value (NAV) and a projected portfolio value by the end of a specified time. The global REIT issues an Initial Coin Offering (ICO), the cryptocurrency industry equivalent of an Initial Public Offering (IPO). Investors interested in offering buy and receive a new cryptocurrency token issued by the company. This token may have some utility in using the product or service the company is offering, or it may just represent a stake in the company or project.

Our use case, a global REIT platform, utilizes three modules deployed to the Hyperledger Fabric network. Each of these modules has a blockchain ledger and demonstrates our implementation of multiple blockchain ledgers. The asset management module consists of an instance of the Resource Access Smart Contract called asset management chaincode and a ledger. It allows asset owners to register their assets with the Fund Manager. The fund manager completes the asset owner's background check and KYC compliance evaluation before the list (launch) of an asset on the platform. Investors have access to assets' information and choose to invest in assets listed on the global REIT platform.

The second module is the transaction module. It deploys a Resource Access Smart Contract called token transaction chaincode. The role of the transaction module is to receive and validates all investors' transaction requests to the platform. The access control module has a Policy Information Ledger and policy graph access chaincode. Policy Administra-

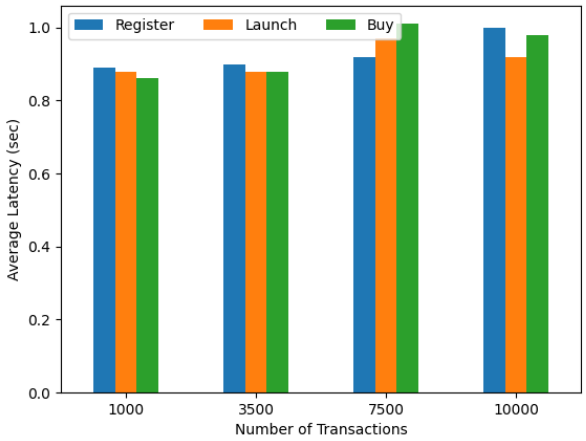

Fig. 6. Average latency of register, launch, and buy transactions using LevelDB

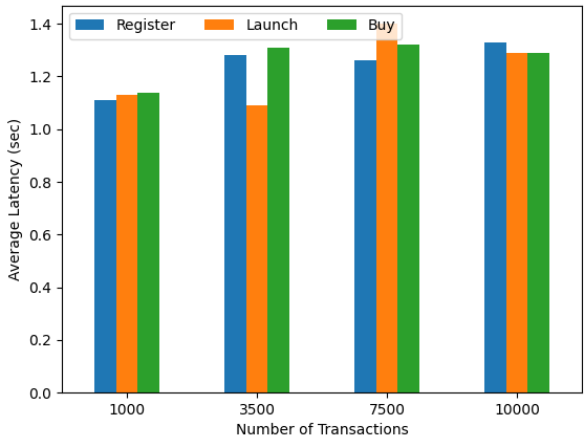

Fig. 7. Average latency of register, launch, and buy transactions using CouchDB

tion and the Policy Decision Smart Contracts constitute the policy graph chaincode. The following subsection explains the graphical representation of the Policy Information Ledger shown in Figure 4

A. Policy Machine Implemented for gREIT

Figure 4 represents a Policy Machine access control graph for the global REIT platform. An algebraic expression for the access graph is G = (PE, ASSIGN, ASSOCIATION). The graph nodes are elements of the set PE, policy element. Nodes on the left side of the graph represent participants (fund manager, asset owners, and investors) and their attributes. The nodes to the right represent the protected (object attributes) assets and transactions. While unlabeled edges (black arrows) of the access graph are elements of the ASSIGN, the labeled edges (blue arching lines) are the ASSOCIATION relations. Excluding the policy class node, gREIT, the assignment (unlabeled edges) from all other nodes must terminate at the gREIT Access node. The policy class is a container that holds all the access rules expressed in the access graph. An element of the ASSOCIATION relation is a triple (user attribute, access right set, object attribute). It implies that a user node with a sequence of assignment (unlabeled edges) that leads to the user attribute of an association can perform the operation(s) granted through access right set on the objects and object attributes that have a path (sequence of assignment) to the association object attribute. For instance, the association (GFM, create, Records) grants the user Admin the access right to perform the action that creates the real estate assets as tokens.

We have utilized the Hyperledger Fabric implementation of the ERC-20 (Etherum Request for Comments 20) for both the token value of an asset and investor's payment for an offering listing on the platform. The ERC-20 is a standard for Fungible Tokens. The records of tokenized assets reside in the ledger for the asset management module, while the record of the purchase of coin offering is the log to the ledger of the transaction module.

B. gREIT Access Request Scenarios

The access rights exchange and buy of the associations (UserIDi, exchange, read, WalletIDr) and (Investors, buy, read, Asset Under Mangt) in Figure 4 models the authorization of purchase by an investor. For an authorized investment in a token asset, an investor must pay from an account (WalletID) associated with the UserID of the investor. Scenario 1 of Figure 5 shows the interaction of the user John with the gREIT platform application. The query request 'readAccount' returns that John has 500 GREIT asset tokens and 2000 ERC20 currency tokens. His authorized 'buyGRET' transaction to invest 150 ERC20 currency in a tokenized asset 'KJKPlaza' response is an asset token balance of 650.

In another scenario, a user (SMBroker) requests to list (launch) an asset for the initial offering service. As the user SMBroker is not associated with the attribute (GFM) granted the access right to launch asset token, the Policy Decision Smart Contract denied the transaction request (see scenario 2 Figure in 5).

VI. Evaluation

We performed this experiment using a virtual machine that has 2 CPUs, 10GB RAM, running Ubuntu 16.04 LTS operation system, and Hyperledger Fabric V2.2 installed. We built a Fabric blockchain testbed which has one Raft orderer service node, two peers for an organization on a single channel. The network configuration is evaluated against the two available data bases (LevelDB and CouchDB).

We generated the three types of transaction workloads into Fabric blockchain using the Hyperledger Caliper V0.4.2. Hyperledger caliper is a blockchain performance benchmark framework, which allows users to test different blockchain solutions with custom use cases, and get a set of performance

test results. Caliper comprises two difference processes, a manager process and scalable worker process. The manager process initialize the Fabric network, schedules the configured rounds, and spools the performance report based on the observed transaction statistics. This experiment, our benchmark configuration sets the number of worker process to 20. For each round of execution ranging from 1000 to 10000 transactions evaluates the average latency for workloads of the transaction types (register, launch, buy).

Figures 6 and 7 plot the experimental results in terms of average transaction latency. The register transaction average latency increases linearly with an increase in transaction number. For 10000 transactions, the average latency for the LevelDB and CouchDB was 1.0 and 1.33 seconds, respectively. The two other types of transactions peaked when the transaction number was 7500. The average latency for the launch transaction was 0.97 and 1.40 seconds for the LevelDB and CouchDB, respectively. Notice a similar trend for the buy transaction. CouchDB results in higher latency than the LevelDB since it incur internal network latency for the required HTTP communication. Conversely, the LevelDB can not effectively support complex schema and queries for the ledger.

VII. Conclusion

In this paper, we proposed the NIST ABAC architecture and its implementation for a permissioned blockchain network. The policy Machine is well-adapted for a distributed system such as the blockchain, and it is scalable.

After we discussed previous work that studies the implementation of ABAC on a blockchain, we presented the system architecture and implementation of the Policy Machine subcomponents as smart contracts (chaincode). We provided a use case to demonstrate the feasibility of the Policy Machine for a blockchain network. The experimental evaluation of this work applies the two available types of databases for Hyperledger Fabric setup, and the Hyperledger Caliper framework provides workloads for average latency results.

A continuation of this work is the study of policy review in a permissioned blockchain network using the Policy Machine. Another known capability of the Policy Machine for future work is applying combined traditional access control policies defined using the Policy Machine for a blockchain network.

Acknowledgment

This work is partially supported by NSF grants HRD-1736209 and CNS-1553696

References

[1] Di Francesco Maesa, D., Mori, P., & Ricci, L. (2019). A blockchain based approach for the definition of auditable Access Control systems. Computers & Security, 84, 93–119.

[2] Mell, P., Shook, J. M., & Gavrila, S. (2016, October). Restricting insider access through efficient implementation of multi-policy access control systems. In Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats, pp. 13-22.

[3] Hu, V., Kuhn, D., Ferraiolo, D., & Voas, J. (2015). Attribute-Based Access Control. Computer (Long Beach, Calif.), 48(2), 85–88.

[4] Hu, V., Ferraiolo, D., Kuhn R., Schnitzer A., Sandlin K., Miller R., & Scarfone K. (2014). Guide to attribute based access control (ABAC) definition and considerations. NIST Special Publication 800-162.

[5] Bocek, T., Rodrigues, B., Strasser, T., & Stiller, B. (2017). Blockchains everywhere - a use-case of blockchains in the pharma supply-chain. 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), 772–777.

[6] Si Chen, Rui Shi, Zhuangyu Ren, Jiaqi Yan, Yani Shi, & Jinyu Zhang. (2017). A Blockchain-Based Supply Chain Quality Management Framework. 2017 IEEE 14th International Conference on e-Business Engineering (ICEBE), 172–176.

[7] Dukkipati, C., Zhang, Y., & Cheng, L. (2018). Decentralized, BlockChain Based Access Control Framework for the Heterogeneous Internet of Things. Proceedings of the Third ACM Workshop on Attribute-Based Access Control, 61–69.

[8] Biswas, P., Sandhu, R., & Krishnan, R. (2016, March). Label-based access control: An ABAC model with enumerated authorization policy. In Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control, pp. 1-12.

[9] Ouaddah, A., Abou Elkalam, A., & Ait Ouahman, A. (2016). FairAccess: a new Blockchain-based access control framework for the Internet of Things. Security and Communication Networks, 9(18), 5943–5964.

[10] Ahlert Pinno, O., Abed Gregio, A., & De Bona, L. (2017). ControlChain: Blockchain as a Central Enabler for Access Control Authorizations in the IoT. GLOBECOM 2017 - 2017 IEEE Global Communications Conference, 1–6.

[11] Qi Xia, Sifah, E., Asamoah, K., Jianbin Gao, Xiaojiang Du, & Guizani, M. (2017). MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain. IEEE Access, 5, 14757–14767.

[12] Azaria, A., Ekblaw, A., Vieira, T., & Lippman, A. (2016). MedRec: Using Blockchain for Medical Data Access and Permission Management. 2016 2nd International Conference on Open and Big Data (OBD), 25–30.

[13] Rouhani, S., Butterworth, L., Simmons, A., Humphery, D., & Deters, R. (2018). MediChainTM: 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), 1533–1538.

[14] Jemel, M., & Serhrouchni, A. (2017). Decentralized Access Control Mechanism with Temporal Dimension Based on Blockchain. 2017 IEEE 14th International Conference on e-Business Engineering (ICEBE), 177–182.

[15] Wang, S., Zhang, Y., & Zhang, Y. (2018). A Blockchain-Based Framework for Data Sharing With Fine-Grained Access Control in Decentralized Storage Systems. IEEE Access, 6, 38437–38450.

[16] Ferraiolo, D., Chandramouli, R., Kuhn, R., & Hu, V. (2016, March). Extensible access control markup language (XACML) and next generation access control (NGAC). In Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control (pp. 13-24).

[17] Ferraiolo, D., Atluri, V., & Gavrila, S. (2011). The Policy Machine: A novel architecture and framework for access control policy specification and enforcement. Journal of Systems Architecture, 57(4), 412–424.

[18] Ding, S., Cao, J., Li, C., Fan, K., & Li, H. (2019). A Novel Attribute-Based Access Control Scheme Using Blockchain for IoT. IEEE Access, 7, 38431–38441.

[19] Pinno, O., Grégio, A., & De Bona, L. (2020). ControlChain: A new stage on the IoT access control authorization. Concurrency and Computation, 32(12)

[20] Zhang, X., Poslad, S. (2018). Blockchain support for flexible queries with granular access control to electronic medicalrecords (emr). 2018 IEEE International Conference on Communications (ICC), 1-6.

[21] Guo, H., Li, W., Nejad, M., & Shen, C. (2019). Access Control for Electronic Health Records with Hybrid Blockchain-Edge Architecture. 2019 IEEE International Conference on Blockchain (Blockchain), 44–51.

[22] Guo, H., Meamari, E., & Shen, C. (2019). Multi-Authority Attribute-Based Access Control with Smart Contract. Proceedings of the 2019 International Conference on Blockchain Technology, 6–11.

[23] Matheus, A. (2005). How to Declare Access Control Policies for XML Structured Information Objects using OASIS' eXtensible Access Control Markup Language (XACML). Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 168a–168a.

[24] Rouhani, S., Belchior, R., Cruz, R., & Deters, R. (2021). Distributed attribute-based access control system using permissioned blockchain. World Wide Web (Bussum).