# DCACI: A Decentralized Lightweight Capability Based Access Control Framework using IOTA for Internet of Things

Sandeep Kiran Pinjala[1,2] and Krishna M. Sivalingam[1]
[1]*Dept. of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India*
[2]*HCL Technologies, Chennai, India*
Email: sandeepkiranp@gmail.com, cs16s001@smail.iitm.ac.in, skrishnam@iitm.ac.in, krishna.sivalingam@gmail.com

***Abstract*—Security and Privacy are some of the important aspects to be considered in the large-scale deployment of Internet of Things (IoT) systems. Due to the large number of IoT devices and the different administrative domains in which they operate, traditional approaches involving a Centralized server for managing Authorizations will not be scalable or efficient. In this paper, we propose a Decentralized Capability-Based Access Control framework using IOTA (DCACI); IOTA is an open-source distributed ledger that enables fee-less micro transactions for the IoT. The DCACI framework enables complete privacy and integrity of the Capability tokens using IOTA's Masked Authenticated Messaging (MAM) technology. It enables device owners and users to Grant, Update, Delegate and Revoke the capability tokens. The proposed DCACI framework has been implemented as a proof-of-concept on a resource constrained machine; the results indicate that it is capable of scaling up to large-scale infrastructure such as a Smart City, having millions of IoT devices.**

***Index Terms*—Capability Based Access Control, IOTA, Blockchain, Internet of Things.**

## I. Introduction

Authorization and access control are some of the primary functionalities required for wide-spread adoption of the Internet of Things (IoT) based systems. Authorization refers to the process of providing the right access to authorized users. Access Control defines how authorization rules are laid out and enforced. As of date, there has been a lack of proper standards for performing device authentication and authorization. Hence, device manufacturers have resorted to *ad hoc* methods to provide these functions. Such methods have lead to numerous attacks against the resource-constrained devices [1].

In IoT systems, enforcing access control for a large number of tiny devices across administrative domains (Federated Access Control, for example in a Smart City deployment) is a challenging task. Each administrative domain might follow its own Authentication and Authorization Policy, which may not be compatible with other domains. Further, there is no common platform for storing and enforcing the policies across domains. Also, there is no centralized authority to verify the integrity of the stored policies. Even if such a centralized authority were to be present, it could become a single point of failure. Lastly, since there is no common platform, ensuring privacy of subjects and objects when enforcing the policies across domains becomes challenging. Having a single distributed database for all the administrative domains would partially solve the above problem. However, there are issues such as maintainability and scalability that need to be addressed.

978-1-5386-4980-0/19/$31.00 ©2019 IEEE

In this paper, we present a Decentralized Capability Based Access Control mechanism using the IOTA system (DCACI). IOTA is a first open-source distributed ledger with fee less micro-transactions for the machine economy [2]. In traditional Blockchain technologies (such as Bitcoin [3] and Ethereum [4]), the rewards of mining a block to the blockchain has become increasingly centralized. On the other hand, IOTA enables the transaction owner to participate in the network by performing a small amount of computational work that verifies two previous transactions. Since there are no miners involved, there are no transaction fees and thus IOTA is fee-free.

The proposed DCACI framework provides inherent security and privacy for the capability tokens stored on the Tangle. The paper presents in the detail the architecture and operations involved in implementing DCACI. The DCACI framework has been implemented in a proof-of-concept system using which performance analysis results are presented. The results help demonstrate the scalability of the framework to large-scale IoT systems, such as Smart Cities.

## II. Background and Related Work

This section presents the relevant background and related work. The Internet of Things (IoT) is defined as a large-scale system built using intelligent and smart sensors and actuators that are connected to Internet. The applications of IoT are in several domains including agriculture, smart cities, industrial environments, health-care to name a few. Everyday items such as bulbs, thermostats, wearable items etc can now send and receive information via the internet. Unlike traditional computers, these devices are highly constrained in terms of the processing power, storage and battery life. Due to these resource constraints, providing end-to-end security and privacy for transactions from these devices becomes challenging. It is also expected that the number of connected IoT devices would be several tens of billions by 2025, requiring highly-scalable solutions. Further, since most of these devices are

available in the open environment, they are highly susceptible to unauthorized access. Hence, incorporating relevant security mechanisms is an important requirement in building IoT systems.

Access Control refers to the process of providing access only to authorized users. Traditional Access Control mechanisms such as Access Control Lists (ACLs), Role Based Access Control (RBAC), Attribute Based Access Control (ABAC) do not scale well to the heterogeneous Operational Technology (OT) environment. Due to the huge number of IoT devices, these schemes suffer from "roles explosion", when the number of resources managed becomes very large [5]. Having a centralized server to manage these rules results in a potential single point of failure. Further, providing contextual and fine grained access control with these traditional approaches requires a larger rule set. In Capability Based Access Control (CapBAC) schemes, permissions are not assigned to users or roles or attributes. Instead, capability tokens are issued to users that uniquely identify the user and the action that the user can perform on the specified resource. Contextual conditions such as place, time of access, etc. can be embedded into the token so that it is valid only when all the conditions are satisfied.

Access Control using Distributed Ledger Technology (DLT) has been recently gaining popularity. In [6], the authors use Bitcoin Blockchain to issue Policy Transactions (PT) and Rights Transfer Transaction (RTT) for policy creation and transfer of rights respectively. The proposal was not specifically designed for constrained IoT devices. In WAVE [7], the authors propose an authorization scheme based on Ethereum Smart Contracts. Delegation of Trust (DoT) and a Protected DoT (PDoT) are defined as edges in a permission graph delegating some permissions on a resource URI. In this scheme, all participating entities have to register their identities with the smart contract and also recursively share the ID private key to their peers for revealing the PDoT. In BlendCAC [8], all participating entities first register with a cloud server that maintains a global profile database. Each domain is managed by a domain coordinator which regularly syncs up the data with cloud server for token updating and retrieval using Smart Contracts. The presence of a global centralized cloud server for managing the profiles and policies can be a single point of failure. Since the tokens are stored on the Blockchain, they are visible to everyone there by raising privacy concerns.

In contrast, the proposed DCACI framework does not need registering any object or identity before the capability tokens are issued. Encryption and signature protection are handled inherently using IOTA's MAM technology, as described later.

In FairAccess [9], the authors use Bitcoin addresses to identify all entities in the network. Similar to Bitcoin's Unspent Transaction Output (UTXO), the UTXO in their scheme is an access token defined by the creator of the transaction to the receiver of the transaction. Unlike the other schemes, FairAccess does not support revocation or updating of Access Tokens. In contrast, the proposed DCACI supports delegation and revocation of Capability tokens and it provides complete privacy and integrity protection of the the tokens stored on the Tangle.

Table I shows a detailed comparison of DCACI with other Blockchain based Access control mechanisms. Unlike other approaches, DCACI operations incur zero transaction fees which makes it ideal for device to device communication in IoT.

## III. IOTA: A PERMISSION-LESS DISTRIBUTED LEDGER

IOTA [2] is a next-generation permission-less distributed ledger that is built on a Directed Acyclic Graph (DAG) called the Tangle [10]. Tangle is not built on blocks chained together as done in Bitcoin and Ethereum. Instead, a single transaction references two past transactions thereby building the consensus. In traditional Blockchain technologies, a small set of miners are responsible for the overall consensus. In IOTA, each network participant is directly involved in transaction approval. Each transaction uses a Random Markov Chain Monte Carlo (MCMC) algorithm to select two random unconfirmed transactions (called Tips) and attaches a proof of work to the transaction. The transaction is then broadcast to the peer to peer (P2P) network using the gossip protocol. An IOTA token is the cryptotoken for all value transactions on the Tangle. All IOTAs which will ever exist have been created with the genesis transaction; also, IOTAs cannot be mined as in Bitcoin and Ethereum.

Some of the main features of IOTA are: (i) There are no transaction fees. Since consensus is achieved without involvement of miners, IOTA does not have any transaction fee. This becomes very significant in case of IoT environments where several tiny sensors exchange data very frequently; (ii) IOTA scales with usability. Unlike other Blockchain based technologies, the more transactions that happen on the Tangle, the better it is for the network; (iii) IOTA allows offline transactions in case some portions of the network do not have reach-ability to internet. The transactions can still continue in the offline partition and when the connectivity is restored, the partition can join back to the main Tangle; (iv) Unlike traditional public key algorithms, IOTA utilizes the Winternitz signature scheme which is believed to be unbreakable even on Quantum Computers.

IOTA's Masked Authenticated Messaging (MAM) [11] enables peers to emit and access encrypted streams of data over the Tangle. The publisher uses a channel ID on which the data stream is published. Subscribers need to subscribe to the channel ID before consuming the data. MAM uses a Merkle tree-based signature scheme [12] to sign the cipher digest of an encrypted message. MAM can be used in three modes based on how the data on Tangle is visible to subscribers. In Public Mode, the Merkle Tree's root is the address of the transaction. Anyone who has knowledge about the address can decode the data. In Private Mode, the Hash of the Merkle Tree's root is used as the transaction address. Unless one has the root, data cannot be decoded. Restricted Mode goes one step further and uses a side key to encrypt the data stream. Unless one has the side key and the root, data cannot be decoded. In

TABLE I
COMPARISON OF BLOCKCHAIN-BASED ACCESS CONTROL MECHANISMS

| | **DCACI** | **WAVE** [7] | **BlendCAC** [8] | **FairAccess** [9] | **Blockchain Based** [6] |
|---|---|---|---|---|---|
| **Technology** | IOTA | Ethereum | Ethereum | Bitcoin | Bitcoin |
| **Meant for IoT Systems?** | Yes | Yes | Yes | Yes | No |
| **Provides Encryption** | Yes | Yes | No | Yes | No |
| **Provides Integrity Protection** | Yes | Yes | Yes | Yes | Yes |
| **Access Control Strategy** | Capability Based | Role Based | Capability Based | Attribute Based | Attribute Based |
| **Distributed Server Architecture** | Yes | Yes | No | Yes | Yes |
| **Capability Delegation** | Yes | Yes | Yes | Yes | Yes |
| **Capability Revocation** | Yes | Yes | Yes | No | Yes |
| **Context Aware** | Yes | No | Yes | Yes | No |
| **Transaction Fees** | No | Yes | Yes | Yes | Yes |

DCACI, we use IOTA's MAM in Restricted mode for storing Capability Tokens on the Tangle. Restricted mode provides inherent privacy to the data stored on the Tangle and only the entities who have the proper side key can decode the data.

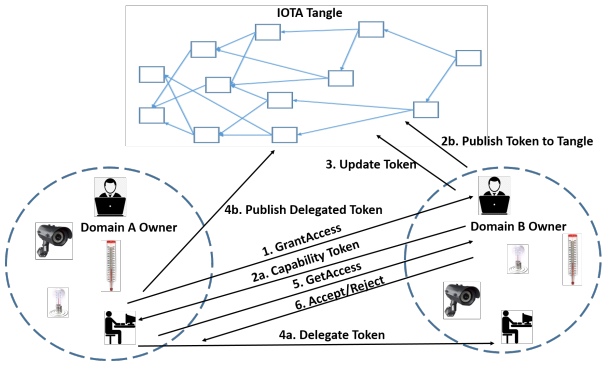


Fig. 1. DCACI System Architecture.

## IV. DCACI: A DECENTRALIZED LIGHTWEIGHT CAPBAC FRAMEWORK USING IOTA

This section presents the architecture and operation of the proposed *Decentralized lightweight CapBAC framework using IOTA (DCACI)* framework. Fig. 1 illustrates the system architecture of DCACI. The framework is intended to be a common platform for enforcing Access Control across various independent domains with no mutual trust established beforehand. Each domain is controlled by an owner, who maintains all the devices in the domain and enforces the Access Control policies within the domain. All accesses to the devices in the domain have to be authorized by the domain owner.

In order to participate in the IOTA network, each domain owner must have a seed. Private keys and addresses used during transactions are generated from this seed. Users (also called as subjects), who wish to access the resources in a domain, must first request for a capability token for the resource(s) along with the action they would perform on the resource. On receiving the request, the domain owner evaluates the local authorization policies before granting the capability token. In DCACI, domain owners maintain these capability tokens on the Tangle. In order to participate in the authorization process, a user need not have an IOTA seed. Only if the user wishes to delegate the access rights to another does it require an IOTA seed. When the user wishes to access a particular resource for which it has been granted a capability token, it presents this token and a request to the domain owner. The domain owner evaluates the request based on local conditions and the latest capability token stored on the Tangle. If the evaluation criteria are satisfied, then access is granted; otherwise, access is denied.

The DCACI framework provides four different operations for enforcing Access Control, as described below.

### A. GrantAccess

When a user in a domain wants to access a resource in the same or in a different domain, it submits a GrantAccess request to the domain owner of the destination domain. It is assumed that a secure communication channel is established between the owner and the user and that the user is properly authenticated before the GrantAccess is requested.

Listing 1. User Request

```
{
  "device": "home/room1/thermostat",
  "rights": [
    {
      "resource": "humidity"
      "right" : "POST"
    },
    {
      "resource": "temperature"
      "right" : "GET"
    }
  ]
}
```

The request is a JSON object that lists all the devices, resources and the corresponding access rights for which the user wants access as shown in List.1. Here, the user is requesting the domain owner for access to device "thermostat". The user is requesting for GET access to the "temperature" resource and "POST" access for the "humidity" resource.

The Domain owner evaluates the local authorization policies against the request for the specific user and generates a

capability token. Based on the local access policies, the owner may cut down on the requested access and/or add additional constraints that need to be satisfied when the the token is presented for access.

Listing 2. Capability Token

```
"id": "6n8rnKdbz3",
"issuer": "owner4",
"subject": "subject0",
"status": "ACTIVE",
"notBefore": "Sun Feb 24 ",
"notAfter": "Fri Mar 01",
"currentAddress": "HPILJFMWY9DQHSGH"
"devices": [
   {
     "device": "home/room1/thermostat",
     "rights": [
        {
          "resource": "temperature",
          "right": "GET",
          "conditions": [
            {
              "time": ">09:00 AND < 17:00
                  "
            },
        }
```

The capability token has a lifetime as specified by the "notBefore" and "notAfter" attributes. Each Capability Token is identified with a unique ID ("id": "6n8rnKdbz3") as shown in List. 2. Once the capability token is prepared, the owner sends the token to the user (through a secure channel) and also publishes it on the Tangle at the address as specified by the "currentAddress" attribute. DCACI uses IOTA MAM in restricted mode to publish the tokens on Tangle. Only the entities having the side key that was used when publishing the data on the Tangle can decode the data back. This provides inherent privacy to the capability tokens stored on the Tangle.

Although the owner has a single unique IOTA seed, it generates a pseudo-random seed for every GrantAccess request based on a combination of the seed and the random ID it generates. It then uses this pseudo-random seed for publishing the capability token (and updating it later) onto Tangle. There are two ways in which the owner can operate. It can send the side key that was used when publishing the token on the Tangle along with the capability token to the user. This way the user can track all the updates that happen to the token at "currentaddress" and see the current status of the issued token. Even though the user has the side key, it cannot modify or create new messages on the MAM stream created by the owner. Only the owner with the right seed can update messages in the MAM stream. If the owner wishes that the user cannot look at the updates to the token on the Tangle, he can send the capability token alone. In cases where delegation capability is requested by the user, the owner sends a HASH of the side key along with the token.

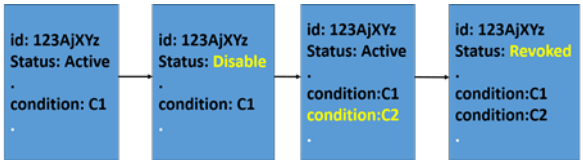


Fig. 2. Token Updates.

### B. UpdateAccess

The *UpdateAccess* operation is performed on a previously issued capability token. It is performed by the domain owner and does not involve the user to which the token was issued. The domain owner may want to update an issued token based on some local condition. For example, change in local policy, capability revocation, suspension etc. The domain owner would publish the updated token as a next message in the MAM stream. As can be seen in Fig. 2, the token has undergone multiple updates over a period of time. The last token in the MAM stream always reflects the current state of the token.

### C. DelegateAccess

Users to whom a capability token has been issued by the domain owner may wish to delegate full or portions of the capabilities to other users. For example, if a contractor has been given access to First floor of a building for renovation, he may wish to delegate access to a room in the First floor to a subcontractor. Delegation is performed only by the user and the domain owner is not involved in the process. The delegated token generation process is similar to the GrantAccess operation. Similar to the parent token, the delegator publishes the delegated token on the Tangle using MAM in restricted Mode. The delegator must have an IOTA seed using which it publishes the MAM data. The side key used for publishing would be the Hash of the side key used for publishing the parent token. Therefore if X is the side key used for publishing the parent token, HASH(X) would be the side key used for publishing the child token.

DCACI uses *SHA256* as the hashing algorithm. As mentioned in the GrantAccess section, the parent may directly send the side key to the child (along with the capability token), or send the HASH of it. When the side key is sent, the child has to use the HASH of it when publishing the delegated token. If the parent sends the HASH of the side key, the child has to use it as-is for publishing data to Tangle.

Listing 3. Delegated Token Attributes

```
    "issuer": "subject0",
    "subject": "subject6",
    "parentId": "6n8rnKdbz3",
    "status": "ACTIVE",
    "notBefore": "Sun Feb 24 2019 ",
    "notAfter": "Fri Mar 01",
    "delegationDepth": 10,
    "currentDelegationDepth": "1"
    "currentAddress": "J78LOPHPILJFM"
```

```
"parentAddress" : "HPILJFMWY9DQHSGH"
```

A token can be delegated only if it has the “isdelegable” attribute set in the original issued token. There is also a maximum delegation depth indicated by the attribute “delegationdepth” that the owner puts in the initial token which indicates the maximum levels up to which the delegation hierarchy can go. The “currentdelegationdepth” attribute indicates the delegation depth of the current token.

List. 3 shows the relevant attributes of a delegated token. Here, “id” is the ID of the delegated token; “parentId” refers to the token from which the current token has been delegated; “subject” and “issuer” have their usual meaning. Further, “currentAddress” is where the token gets published on the Tangle and “parentAddress” is where the parent token is published.

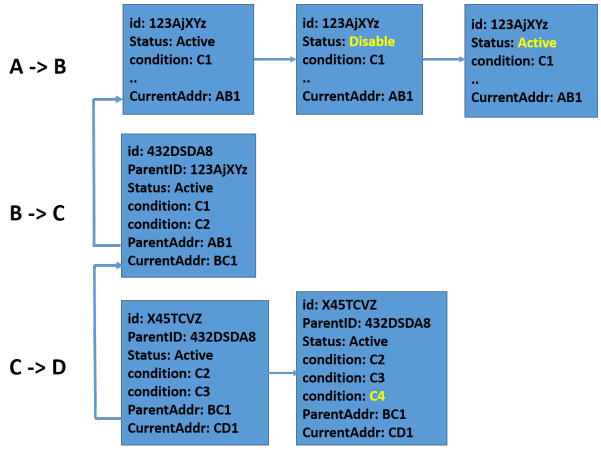


Fig. 3. Token Updation and Delegation.

Fig. 3 shows both the token updation and delegation. Here, A issues the token to B, B then delegates it to C, who then delegates it to D. A and C update the tokens after it was issued.

### D. GetAccess

The *GetAccess* operation is invoked by the user when it wants to perform a specific operation on a resource for which it was earlier granted a capability token. The user submits the request it wants to perform along with the capability token that vouches that the user has the right to perform the requested action on the resource. As mentioned earlier, it is expected that the user submits the request to the domain owner over a secure channel. The domain owner would want to verify that the user is indeed the one for which the capability token was issued (“subject” attribute in the presented token). The presented token could be the one directly issued by the domain owner or one that has been delegated across multiple levels. The owner runs the “PathBuilding” logic to construct the hierarchy of tokens in a bottom up fashion ending with the one that was issued by the owner.

As can be seen in Fig. 4, the owner does this by following the “parentaddress” attribute through the delegation chain. Since the side key used at each level in the delegation

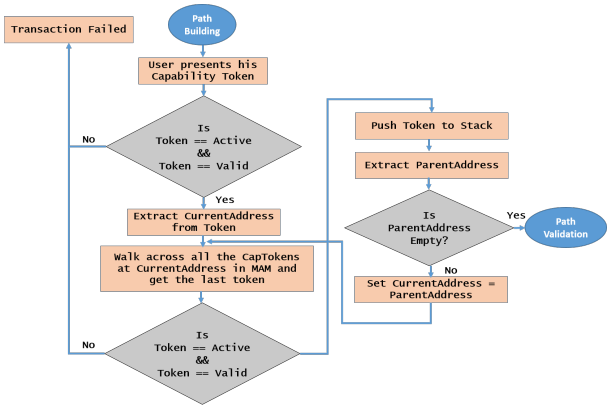


Fig. 4. Token Path Building Logic.

hierarchy is the HASH of the previous level and the domain owner holds the side key with which the first token was issued, it can extract all the tokens in the hierarchy.

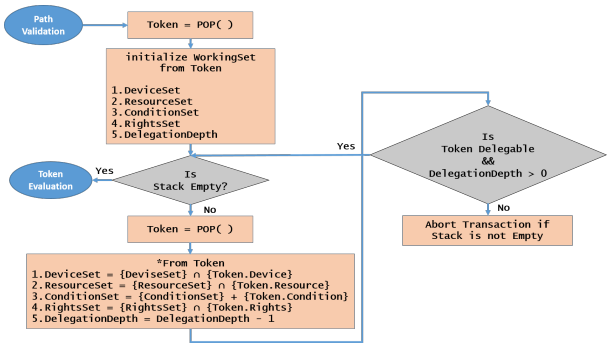


Fig. 5. Token Path Validation Logic.

Once the owner builds the complete token hierarchy, it runs the “Pathvalidation” logic (shown in Fig. 5) in a top down fashion starting with the token that it issued. The current DeviceSet, ResourceSet, RightsSet etc are an intersection of the previous working set and the ones extracted from the current token.

The working set after the “PathValidation” operation completes is then evaluated against the user request. If the devices, resource, rights and constraints are all satisfied, the user is granted access to the specified resource. If any error is encountered at any point in “PathBuilding” or “PathValidation” or evaluation process, the transaction is terminated and the user request is rejected.

## V. Implementation and Analysis

A concept-proof prototype has been implemented and tested against the IOTA Tangle. The implementation includes a daemon service written in Node.js for the “Domain Owner” that continuously listens for requests from Users. The owner is in charge of a set of IoT devices comprising of a set of resources and certain actions pertaining to those resources. Users can make requests for resources upon which the domain owner would evaluate local policy and grant/reject the request.

On success, a Capability token is issued to the user which it presents back to the owner when it wants to access a particular resource. User requests are fired from a Node.js application that submits the GrantAccess and GetAccess requests. Users can also delegate capabilities to other users. The owner's daemon service periodically updates the capability tokens that it has issued to the users.

The domain owner service was run on a Raspberry Pi 1 node with quad-core ARM v7 CPU and 1GB RAM running Raspbian GNU/Linux 9 OS. The owner service was connected to IOTA Public node *https://nodes.thetangle.org/*. User requests for GrantAccess and GetAccess were generated from two systems with 1GB RAM, 1 CPU 2.4 Ghz Intel(R) Xeon(R) CPU processor running Ubuntu 18.04.1 LTS. A separate application was run on the Raspberry Pi node to randomly update the issued capability tokens. In order to support delegation, the user system was also running the same domain owner service.

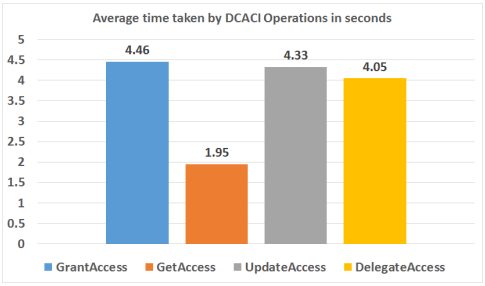


Fig. 6. Time consumption of DCACI operations.

Fig. 6 presents the average time taken by each of the DCACI operations over 500 such transactions. Most of the operations took less than 5 seconds, with GetAccess requiring less than 2 seconds. Table II shows the finer details of the time consumed by the DCACI operations. As can be seen, the majority of the time is consumed during storing and retrieving data from the Tangle.

TABLE II
DETAILED TIME CONSUMPTION OF DCACI OPERATIONS.

| Operation | Time (in ms) |
|---|---|
| **GetAccess** | |
| Path Building | 2.5 |
| Path Validation | 0.03 |
| Token Evaluation | 0.005 |
| Fetch From Tangle | 2,281.714 |
| **GrantAccess** | |
| Grant Access | 264.657 |
| Attach to Tangle | 2,979.671 |
| **UpdateAccess** | |
| Update Access | 325.634 |
| Attach to Tangle | 3,131.821 |
| **DelegateAccess** | |
| Delegate Access | 263.102 |
| Attach to Tangle | 2,550.734 |

In IOTA, the Proof Of Work (POW) is attached to the transaction before it is published onto the Tangle. It is this POW by the IOTA node that takes considerable portions of time in Grant, Update and Delegate operations as they involve publishing data to Tangle. This is the reason the DCACI operations are in the order of seconds. In Bitcoin and Ethereum, POW is done by the miners once sufficient transactions can be blocked together into a block. And each block is mined every 15 seconds on an average in Ethereum and 10 minutes on Bitcoin. Unless a transaction appears on a block, it cannot be treated as validated by the Blockchain network. So although the DCACI operations look to take a longer time, they perform better than Bitcoin and Ethereum based solutions.

## VI. Conclusions

In this paper we proposed a decentralized capability based access control mechanism for IoT (DCACI) based on IOTA's Tangle. DCACI uses IOTA's MAM technology to publish capability tokens on Tangle. DCACI supports Grant, Get, Update and Delegate operations with inherent privacy protection. A concept-proof prototype has been built with multiple owners and users running on different systems. Simulation results show that our proposed mechanism provides a scalable and fine-grained access control mechanism for IoT networks. One possible extension to the current work is to study how authentication and channel security can be provided in this framework.

*Acknowledgments:* This work was supported by a DST grant (EMR/2016/003016) from Government of India (2017-2020) and an IRDA grant from IIT Madras (2017-2020). The authors thank Dr. Chester Rebeiro of IIT Madras for his valuable comments on the paper.

## References

[1] Vanniarajan Chellappan and Krishna M. Sivalingam. *Internet of Things*, chapter Security and Privacy in the Internet of Things: A Survey(Edited by Rajkumar Buyya and Amir Vahid Dastjerdi). Elsevier, 2016.
[2] IOTA. https://www.iota.org.
[3] Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/en/bitcoin-paper.
[4] Ethereum. https://www.ethereum.org/.
[5] Sergio Gusmeroli, Salvatore Piccione, and Domenico Rotondi. A capability-based security approach to manage access control in the internet of things. *Mathematical and Computer Modelling*, 58:1189–1205, 09 2013.
[6] Damiano Di Francesco Maesa, Paolo Mori, and Laura Ricci. Blockchain based access control. In *Distributed Applications and Interoperable Systems*, pages 206–220. Springer International Publishing, 2017.
[7] Michael P Andersen, John Kolb, Kaifei Chen, Gabriel Fierro, David E. Culler, and Raluca Ada Popa. Wave: A decentralized authorization system for iot via blockchain smart contracts. Technical Report UCB/EECS-2017-234, EECS Department, University of California, Berkeley, Dec 2017.
[8] Ronghua Xu, Yu Chen, Erik Blasch, and Genshe Chen. Blendcac: A blockchain-enabled decentralized capability-based access control for iots, 2018.
[9] Aafaf Ouaddah, Anas Abou Elkalam, and Abdellah Ait Ouahman. Fairaccess: a new blockchain-based access control framework for the internet of things: Fairaccess: a new access control framework for iot. *Security and Communication Networks*, 02 2017.
[10] IOTA Tangle. https://www.iota.org/research/meet-the-tangle. [Online; accessed Feb-2019].
[11] IOTA MAM. https://blog.iota.org/introducing-masked-authenticated-messaging-e55c1822d50e.
[12] Merkle Trees. https://www.imperialviolet.org/2013/07/18/hashsig.html.