2016 IEEE First International Conference on Internet-of-Things Design and Implementation

# World of Empowered IoT Users

Sayed Hadi Hashemi, Faraz Faghri, Paul Rausch* and Roy H Campbell
*University of Illinois at Urbana-Champaign, *ExaByte Labs*

***Abstract*—In a world deploying an Internet of Things, sensors and actuators are owned, accessed, and activated by a plethora of individuals and organizations. Access to the data produced by this world can both be beneficial and have drawbacks to society. This data potentially represents the activities of millions of individuals and their possessions collected by billions of "things'. Aggregations of this data can be analyzed through the Internet and Clouds. This raises possible privacy, security, moral and ethical challenges whose solutions will require flexible protection mechanisms. How do we "acquire" and "distribute" data at the IoT world scale while retaining the rights of individuals and organizations to protect, use, and share their data? Clearly a well-defined mechanism and control needs to regulate access to the data and its aggregations.**

**Our paper describes a user-centric multi-level multiple granularity mechanism to share the data from these devices to people and organizations. Revisiting the fundamental mechanisms in security for providing protection, our solution uses capabilities, access lists, and access rights following well-understood formal notions for reasoning about access. Our contribution is to describe an auditable, transparent, distributed, decentralized, publication-subscription based, robust mechanism and automation of these ideas in the IoT realm that is well-matched to the current generation of clouds. It is based on well-tested principles and practices used in crypto currencies exploiting block chains of transactions. The scheme puts users (including organizational entities) in the center of control over the access to their collections of sensory data. In our paper, we describe a deployment of these ideas for health care, smart cities, and autonomous cars.**

## I. Introduction

The Internet of Things is growing and sensors are being installed at ever increasing rates. In not too long a fully deployed IoT will become a reality, with it comes huge promises for the future of Big Data, Analytics, and research in general. However, this promise relies on accessibility of massive amount of data which resides on the clouds and at the edge (e.g. sensors) and owned by IoT users. Our current models for data sharing and user privacy will not scale to this level. In order to meet the expectations of this revolution, data needs to be controlled by users and shared easily with multiple parties, but this raises new challenges.

Scalable data sharing is already becoming an active problem in several areas of IoT including healthcare, smart cities, and autonomous cars. For instance, intelligent decision making in these areas involves analyzing sensitive data in heterogeneous environments. Health Tracker data is providing a particularly difficult challenge as Electronic Health Record (EHR) systems attempt to integrate more and more patient data from devices that are bridging the gap between consumer electronics and medical devices. Gaining access to this data, while keeping the user informed and empowered is proving difficult. The same can be said about the huge number of sensors in GPS devices and phones that remain siloed and inaccessible due to concerns over data privacy and sharing. In these environments, users themselves frequently do not know what private data has been stored, who manages it, or where it resides.

Data sharing provides both huge benefits and presents substantial risks for society. We should be able to provide previously unimaginable insights and a level of control unequal to anything prior. The substantial benefits make it worth overcoming the issues of sharing and meeting the IoT's goal of providing an extremely high level of autonomicity and device intercommunication.

Arguably, there is always a user (or organization) who owns or manages data and this user factor should be the ultimate deciding factor in regard to the data sharing. What we need is a system that can securely provide this basic functionality for locating and sharing data between things, and/or entities. However, as IoT scales to billions of sensors and millions of users, conventional sharing and access control solutions do not scale. It becomes impossible to maintain an Access Control List (ACL) on each single sensor, or near impossible to rely on a centralized access control server, or agreed upon trusted parties based protocols. Networks of billions of devices and millions of users would require maintaining enormous accessible access control lists that if possible at all would be very expensive and challenging with their strict availability and consistency needs.

A protective mechanism tailored for this setting which enables sharing and eliminates (or reduces) risks is necessary. In this paper we present a model to address these challenges utilizing best practices in computer security and distributed systems. Our solution is to build a user oriented data dissemination and distribution system. Data collected by the IoT potentially represents the activities of billions of individuals or organizations and their possessions, it is intuitive that we should put these data owners in control of their data rather than third parties. Naturally, trusted third parties should contribute in helping ascertain the trustworthiness of a party. But this role should be advisory, whereas now they are actively making these decisions on behalf of data owners.

Our system has three main components, the first is a data management protocol in which different roles can interact with each other in a secure and private manner based on capability-based access control. The second is a decentralized and distributed data store system that allows each role able to communicate with each other without the

978-1-4673-9948-7/16 $31.00 © 2016 IEEE
DOI 10.1109/IoTDI.2015.39

need for a trusted third party based on Blockchain. Although our solution has a different purpose, recent cryptocurrencies such as Bitcoin provide an interesting and similar successful implementations of decentralized and secure circulation of currency. The third is a scalable messaging service based on publish-subscribe model designed to provide flexible and reliable communication between many senders and receivers without the need of persistent connectivity. Our solution is fully decentralized and distributed in order to accommodate the massive scale of the IoT paradigms. Because the system is user-centric, no trust of any participant is implied and automatic. While still maintaining the ability to scale, this provides full control of access control and propagation, system security, and user privacy.

The remainder of the paper is organized as follows. In Section II, we give an overview of how IoT plays an important role in three emerging domains and fundamental data exchange challenges they are facing. The overview of our solution is the subject of Section III. Sections IV, V, and VI will describe the three components of our solution, data management protocol, data store system, and messaging service respectively. We discus the practicality of our proposed solution and future works in Section VII. We provide related works in Section VIII, and conclude in Section IX.

## II. IoT Worlds

IoT finds its way to enhance our society in many domains. These systems are tangled with users' every day job. In this section, we provide an overview of how IoT embodies in different applications, and what are the open challenges. All these applications include billions of data sources including sensors and millions of users. For the benefit of each application, data has to be shared with data analyzers.

### A. Health Care

The amount of data generated by healthcare providers is increasing rapidly [1]. Much of this new data is being generated by a new generation of things. Medical devices are now communicating directly with Electronic Health Record (EHR) systems. When we look at these trends and consider them in the context of the evolving discussion regarding patient privacy and data security, our model presents some unique opportunities to facilitate better sharing of data between parties.

In the past patient records were typically stored on paper in the offices of their physicians. Most countries provide a legal mechanism requiring physicians to archive records for a certain period of time, as well to provide patients access to their medical records. However, when it comes to ownership of the records themselves, as well as to when they may be shared for research purposes, there exists no such consensus. Because data is also stored with a wide range of providers and organizations, patients may not even be aware of all of the places where their data is stored. This provides challenges for protecting that data, as well as providing access to it. Our system enables healthcare providers to share electronic health records directly with each other, and avoids the need for information exchanges and paper authorization forms.

It is becoming clear that patients need better privacy protections. This is happening at a time when the demand for this data from bona-fide research endeavors is also increasing. This creates a mismatch whereby more protected data is necessary at a time when patients are pulling back on what they want to share. Our model addresses this by providing more power to the patient and facilitates the inclusion of a larger patient population. Furthermore, our model can reduce liability and depoliticize when it is appropriate to share records as each patient is able to opt-in or out at their discretion. Currently much of this data may be protected by complex competing policy constraints or overlapping IRBs (Institutional Review Board). Although these are good practices when the patient cannot be directly involved, they provide significant hindrances if current research efforts are to scale. Our model also provides a particular advantage in countries which proscribe specific requirements and liabilities upon organizations that hold or manage healthcare data. Because our system does not store any data, or user identifiable information, it may not be subject to the regulatory requirements of entities covered by healthcare privacy regulations such HIPAA, Directive 2011/24/EU, and other similar laws.

### B. Smart City

The promise of the IoT and Big Data Analytics for future cities is the reduction of frictions and inefficiencies. By installing sensors and effectively analysing data, there are numerous opportunities to revolutionize many aspects of city management. Public transport, public health and safety, energy management, infrastructure, and environment will all evolve to take advantage of this data. Sensors collect data regarding the behavior of passengers, for example how denizens commute to work. Public transport systems could aggregate data, make intelligent decisions, and convey actuator orders. This would allow for extremely efficient utilization of transportation resources.

Without Big Data Analytics the IoT cannot produce actionable insight. For instance the municipality reroutes services based on demand by conducting the analysis of their ridership data. But in order to take full advantage of the IoT, data will need to be available for analysis to a diverse group of external sources. Cities will need to involve external researchers like social entrepreneurs, private sectors companies, startups, in additional to more traditional participants like city departments, healthcare providers, and academia.

As these agencies and governments deploy sensors we find that this data is owned and managed by such a wide range of organizations that data becomes effectively siloed. Furthermore, private companies, individuals and NGOs may wish to collaborate with governments. Imagine a network of smart homes, or building automation systems working directly with electric producers. This is further complicated by a wide range of laws concerning public records and individual privacy. In order to make the dream of smart cities

a reality, it will become critical to have more flexible and scalable data sharing models to allow for inter organization sharing.

Our model allows for such sharing at both small and massive scales. Imagine a group of cities that have agreed to work together with a group of universities to provide detailed data regarding law enforcement. Many cities already have surveillance cameras, body cameras, drones, gunshot detection equipment, GPS fleet management, as well as a massive array of sensors that are yet to be developed. However, this data is and will be owned and managed by municipal police, regional police, municipalities, as well as many layers of government. The current centralized models would not allow a large group of researchers to gain access to these highly heterogeneous and disparate data sources. Also, they would create significant delays as well as requiring full time employment of an entire department of administrators and supporting administrative personnel. Once deployed, our model can effectively handle this task quickly, and securely.

### C. Autonomous Cars

Autonomous cars have moved from being prototypes with a massive array of unwieldy instruments into mature and polished products with millions of real world miles. It is likely within the next 5 years that many cars will be able to gather data and communicate with each other and the Internet.

Autonomous cars are going to require data communication with both their surrounding cars, as well as with centralized traffic management. Cars will need to communicate securely data such as speed, braking, signaling and collision avoidance data. These are all prerequisites to enable autonomous cars to share roadways at high efficiency.

Currently, no one company has a comprehensive view of these problems. Many cars already have a mechanism for communication (OnStar and similar services). Some manufacturers are also beginning to take this problem seriously and conduct research into car to car communication technologies [2]. Recently, GM has made a significant investment in Lyft in order to build a network of self-driving cars and align itself strategically when ride sharing services evolve into autonomous car sharing services [3]. Ride sharing services themselves are also aggregating substantial usage data. Right now this data is becoming siloed as car owners have little control over who has access to them.

However, all of this ridership, usage data and car to car data could be quite sensitive. Furthermore, sensor data on cars will become integrated with calendars and intelligent personal assistants like Google Now in order to provide an integrated experience. Our system allows for autonomous cars to decide what, when and with whom their data is shared.

## III. World of Empowered IoT Users

As the IoT grows and matures, the number of connected 'things' is rapidly increasing. There is a growing need to share the data produced by these things. However current security models do not work well for the complex needs of the heterogeneous systems emerging in the IoT. Previous systems often placed data on equipment owned by the same owner as the data itself. Within the emerging IoT often times data is stored in a variety of locations, managed by different service providers, and subject to varying regulations. These systems less and less frequently have centralized control, and are typically managed by a very diverse group of service providers. Because of these heterogeneous and disparate environments it is often difficult to manage the issues of trust relationship management, data management, regulatory compliance and data sharing.

The IoT solutions being employed today either sacrifice scalability or user control. Google serves more than one billion users globally, arguably, offering the most scalable products on Earth. The issue of Google storing users' personal data has been 'addressed' by a clause in the privacy policy. Users are commonly unaware of ways their data is stored and managed. This confusing wording reached the attention of regulators, eventually culminating in Google's fining by the European Union [4]. Meanwhile, at 23andMe, the largest consumer based genomic company, the issue of privacy delegation surfaced. Users had delegated the management of their data to 23andMe who had subsequently sold the data [5]. In both cases the user has lost control of their privacy, and are no longer actively able to decide what should be shared.

Granting control to users at the massive scale of IoT is a very challenging problem. At the scale of IoT, conventional best practices are not practical, it becomes impossible to maintain an access control list on each single sensor, or near impossible to rely on centralized access control server, or an agreed upon trusted parties based protocols. Server-client models such as Kerberos [6] can not be used in these environments due to lack of central point of trust. Distribution and privacy concerns make implementation of very basic Big Data Analytics primitives nearly impossible. For instance, in the current setting, 'dataset lookup', the simple task of finding a data set is a major quest.

Access control at massive scale is not the only challenge. Solutions have to still overcome many traditional IoT challenges including connectivity. IoT sensors might not be nearly as well connected. An example of this might be the network of sensors implemented in a house that is not necessarily exposed to the whole world.

Our solution's goal is to provide a mechanism for the user to control at scale. Two differentiating contributions in our solution are: first, separation of the data store from the data management, and second, architecting both components scalable, decentralized and distributed. These two major contributions provide substantial benefit for a user-empowered IoT; it grants the possibility of keeping data at its origin, overcomes the single point of trust and failure, and adapts to the growth of users and sensors. Before explaining the solution in depth, we provide an overview of our IoT world. Further, we explain the roles, primitives offered to these roles, along with the scalability and threat assumptions

which are considered in this world. For the rest of this paper we use Abadi et. al notations [7], [8] (explained in I) to formally describe the system and protocols.

### A. Roles

Four main roles are introduced in our world:

- **Data Owner:** an individual or organization who is in possession of data. This role does not necessarily generate or store the data. In our system, data owner grants the access to the data.
- **Data Source:** represents a computer system, individual, or organization, who manages and stores data objects, be it at rest or in motion. A sensor can act as data source if it has enough performance, connectivity, and storage, otherwise its data is stored elsewhere. Examples would include cloud providers, managers of EHR systems, application gateways, and archival systems.
- **Data Requester:** an individual or organization that requests access to other data owners' data, available within the network. Examples are researcher, company, data aggregator, or another device.
- **Endorser:** a third party individual or organization that validates a request. This may be a trusted authority, organization, or known individual. Endorser either provides supplementary information regarding credibility, or validates authenticity of the role's identity. Examples of endorser are but not limited to:
  - Internal Review Board (IRB)
  - External review by additional research institution (Hospital A asks hospital B for review)
  - Governmental entity (e.g. FDA, NIH, municipal, state health department)

### B. Primitives

Following primitives are considered for the above roles in our world:

- **Data Discovery:** users have many sensors and more likely no substantial data storage. They should know what data they own or have the rights to. They should have one view that allows them to see a portfolio of all of the data they own.
- **Data Request:** the subject should be able to search for a collection of data that meets their conditions. The request should be easily converted to a data access authorization.
- **Audit:** users should be able to share their data with any chosen data requester, as well as track and monitor the requester's access.

### C. Assumptions

Following scalability and threat assumptions are considered in our world:

A1. We assume that data sources are aware of their users' identity (public key), i.e. data source $S$ knows its user's public key $K_X$, and can securely authenticate signed messages by that user:

$$\{message\}_{X'} \implies X \text{ says } message$$

Table I
NOTATIONS USED IN THIS PAPER ADOPTED FROM ABADI ET. AL [7], [8]

| Notation | Description |
|---|---|
| $\{M\}_X$ | Encrypted message $M$ using $X$'s public key. |
| $\{M\}_{X'}$ | Signed message $M$ using $X$'s private key. |
| $X$ | Identity of $X$. |
| $K_X$ | $X$'s public key. |
| $O_i$ | Data Owner $i$. |
| $S_i$ | Data Source $i$. |
| $R_i$ | Data Requester $i$. |
| $E_i$ | Endowser $i$. |
| $DOT$ | Data Object Ticket. |
| $DAP$ | Data Access Path. |
| $RT$ | Request Ticket. |
| $DAT$ | Data Access Ticket (Capability) |
| $X$ *says* $Y$ | $X$ makes the statement $Y$. |
| $X$ *for* $Y$ | $Y$ on behalf of $X$. |
| $X$ *controls* $Y$ | $X$ is trusted on $Y$: if $X$ says $Y$ then $Y$. This is the meaning of $X$ appearing in the ACL for $Y$. [7] |

A2. We assume that during user registration, users are provided with a mechanism to authenticate the identity of the data source. This mechanism can be verification from a certificate authority or a token:

$$\{K_C\}_X \implies C$$

A3. We assume that the endorsers' identity can be relied upon as valid via currently available technologies such as PKI, i.e. if a subject trusts a certificate authority such as $C$ which signs an endorser identity $E$, then:

$$K_C \implies C$$
$$K_C \text{ says } (K_E \implies E) \rightarrow K_E \implies E$$

A4. We address the issue of traffic analysis in this paper; however we make a best effort attempt rather than attempting to completely obfuscate users' access patterns.

A5. It is assumed that users may trust their local computing environment. We understand that this assumption is difficult to guarantee and the execution of sensitive programs in untrusted environments will continue to be a risk.

A6. We assume that attackers have a specific set of abilities. We assume that attackers can view system's data, are able to present modified data to participants, and may impersonate roles. We also assume that attackers do not have access to private keys, and cannot gain local administrator access to the systems.

A7. We assume the security of the Blockchain system, and cryptographic integrity of the utilized encryption protocols.

A8. We assume that 'things' either have the capability to act as a data source or are able to transmit their data to a data source securely.
A9. We assume that each data object has at least one data owner.
A10. We do not address DoS attacks in the current system. In the future works, this attack can be addressed by the best efforts and practices.

### D. Our solution

Reviewed the roles, primitives, and assumptions in our world, we introduce our solution for empowering the IoT users. Our solution's goal is to provide a mechanism for the user to control the access to their data at scale. Our system consists of three main components:

1) Data management protocol
2) Data store system
3) Messaging service

An overview of our system with the three components is illustrated in Figure 1. Data management protocol provides a framework for interaction among different roles with an access control mechanism. The protocol is a secure capability-based system. Data store system provides a persistent, distributed, and decentralized storage for the access control component; it is based on Blockchain and provides full transparency of transactions very similar to Bitcoin. The messaging service plays an important role for the scalability of our solution; it is based on publish-subscribe architecture and designed to provide scalable, flexible, and reliable communication between many senders and receivers without the need of persistent connectivity.

Our solution is user-oriented; users are the bridge between different networks of things, and they are ultimately authorizing the access to their data. Information regarding the user's possessions is sent to the user securely and privately. This data is accessible whenever the user demands to view it. Data owners are aware of the existence of data by direct knowledge or control over the sensor/actuator producing the data. Additionally, they may learn about the existence of data via notifications generated by the data sources and transmitted directly to the data owner, mainly to communicate the management of data by the data source.

Data requests are broadcasted to all the data owners within the system. Data owners are not required to review all the requests. In case a data owner is not interested in sharing any data, they can easily filter out all the requests, or they can selectively filter out requests based on their interest. Also, User's trusted endorsers may verify a data request. This helps to protect users against malicious or unethical data requests, as well as assisting them assess the risk of received data requests.

Data requests, upon arrival by the user, will be checked against the user's portfolio in the client. If the conditions are met, and data owner is willing to share the data, a capability ticket will be issued to the data requester which provides necessary authorization for accessing the requested data.

Whenever two roles want to interact, one sends a message to the messaging service component. Then, the elements of messaging service will send the message to the data store system which is based on Blockchain. After successful storage of the message in the Blockchain (data store system), it will be fetched by the recipient through the messaging service.

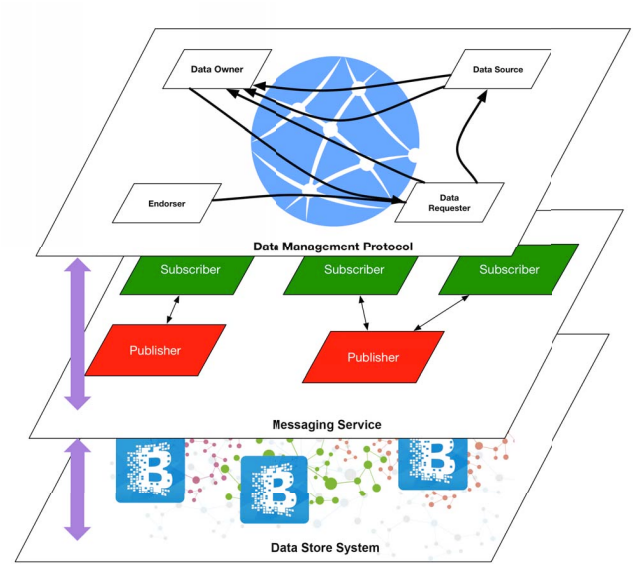


Figure 1. Our solution consists of three main components: *i.* data management protocol, *ii.* data store system, and *iii.* messaging service. Data management protocol is a secure capability-based system which provides a framework for interaction among roles. Data store system based on the Blockchain provides a persistent, distributed, and decentralized storage for the access control. The messaging service based on publish-subscribe architecture provides scalable, flexible, and reliable communication between many senders and receivers.

In the next sections, we explain our solution's three main components in detail.

## IV. Data Management Protocol

Data management protocol provides a framework for different roles to interact in a secure and private manner. The protocol is a decentralized capability-based access control. The communication between different roles is performed by message passing. Each message contains a sender, a receiver, and a payload. Messages are delivered through the underlying "Messaging Service" which is described later in Section VI.

In this section, first, we compare our data management protocol to capability-based access control systems. Later, we introduce data structures used in the message payloads along with the messages transmitted in the protocol. Through this section, we use notions described in Table I.

### A. Capability-Based Access Control

Our system is implementing a distributed and decentralized capability based access control system. Capability is a token that gives a data requester permission to access a data owner's data object on a data source. Using capability based access control, our system avoids the necessity of having a

centralized trusted party to confer trust. Instead, the data owners are responsible for issuing the capability to other users.

In our implementation, *capabilities* are issued by the data owners ($O$). Each one indicates who is the issuer, to which data requester ($R$) this capability is issued, and the object to grant access to. Additionally, each capability contains the access rights in the form of data queries, it also contain information about where and how the data can be accessed. Capabilities are not transferable or usable by any other subject rather than intended data requester. As the result, the data owner will remain the sole controller of the data:

$$(R \textit{ for } O) \textit{ says } d$$

Our capability is implemented as a data structure that contains:

- **Access rights:** every capability issued contains a query that adds additional restrictions on the data access. In the case of data, restrictions are limitations on the access to data attributes, elements, or may apply a accumulative function on the data. In the case of actuators, a query can restrict the types of actions to be sent.
- **Identities:** they are used to describe uniquely subjects of a capability. Due to the decentralized nature of the system, it is not an easy task to identify every subject. Therefore, public keys are used to identify subjects in the capability. Based on implementations such as RSA, or PKI, it is assumed that generated key pairs are always unique. Capabilities are issued and used to only represent one public key (one identity). Multiple entities can share a public/private key pair in order to implement joint ownership.

$$K_X \Rightarrow X$$

In our implementation of capability, following operations are contained:

- **Create:** right to create capabilities is restricted to the data owners ($O$). As soon as the data object ($d$) is generated and stored on a data source ($S$), this right is delegated to the owners of data to give access to possible data requesters ($R$). This creation of right is not transferable.

$$\forall R, O \textit{ controls } [(R \textit{ for } O) \textit{ says } d]$$

- **Delegate:** deleting a capability is not possible for the sake of audition and non-repudiation. However, a data object referred by the capability can be revoked by moving (or removing) the data on the data source.

**Audit** is a critical feature to track who, when, and where has access a data. In the absence of a centralized trusted party, our implementation provides the audit feature through the capability call back key. Data sources will inform the data owners when capabilities are used to access data objects.

Also, the capability is stored by the data requester. Upon arrival, capability can be verified without an external *access control list*. As a result, having more capabilities or data objects may not require extra storage or computing power on the data storage or the data owner.

### B. Protocol Data Structures

Five data structures are used in the payload of messages. *Data Objects* are data at rest or data in motion stored in the data sources. Even though the data objects are not directly used and transmitted in the message, the ultimate goal is to manage their access. Data objects could be files or records within a file repository, database, cloud provider, or any other data storage system. In practice, there are no limiting factors that would prevent our system from supporting other documents such as paper documents, film, recordings or another non-digitized media. Data Objects in motion could be a variety of live data sources, including but not limited to: sensors, video streams, audio streams, health trackers, location data, etc. Lastly, a Data Object may represent an actuator or device that the data requester may interact with. An example may be a Physical Access Control Systems (door locks, and alarm systems), a PTZ camera, a traffic light, a shared vehicle ignition system, etc.

Five data structures used in the payload of messages transmitted in the Data Management Protocol are:

1) **Data Object Ticket (DOT):** the right to issue a capability to access a data object. This ticket is issued by the data source and contains the unique id of specified data object, the identity of the owner ($K_O$), the identity of the data source ($K_S$), data access path ($DAP$), and metadata describing attributes of data object, signed by the data source:

$$\text{DOT} = \{\text{Data ID}, K_O, \text{metadata}, \text{DAP}\}_{K'_S}, K_S$$

2) **Data Access Path (DAP):** this message represents a mechanism for an authorized role to gain access to the data object. Some examples of this DAP can be:
   - URL/URI (FTP/SFTP/SCP/HTTP/HTTPS)
   - Record Locator
   - Contact Information
   - Instructions
   - Physical Location

   Note: In our implementation, tickets can be utilized as the authentication mechanism for TLS/SFTP/SSH as they are generated from the TLS standard.
3) **Request Tickets (RT):** these are messages broadcasted by the data requester to all the subscribers. A data request ticket typically consists of a data query, participation conditions, duration of access, and some relevant metadata:

$$\text{RT} = \{\text{Request ID}, K_O, \text{Query}, \text{Conditions}, \text{Duration}, \text{Metadata}, K_R\}_{K'_R}$$

   A data query indicates to the data owner what data the data requester intends to collect from data owners. The query format is determined by data sources.

Participation conditions are constraints on data owner in order to determine if they are qualified to participate in the data request. For example in a medical data collection this condition can be on:

- Nationality of Participant
- Demographic information about participant, age, gender, or residency
- Relationships with certain data source providers, i.e. only data stored with Google or Apple

4) **Endorsements:** provide a method for an endorser or endorsers to provide additional third party information to the data request. This information will help the data owners to decide whether they are willing to share data for a request. For example, endorsement can be:
   - Metadata regarding the data privacy policy of the data requester
   - Results of a third party audit or attestation regarding the information security controls in place at the organization
   - Results of an Internal Review Board (IRB) decision
   - Authorization by a government entity to conduct a trial or research (FAA, NIH, HHS)
   - Indication that data will be shared in a joint collaboration
   - Verification of the data requester authenticity by a service provider or other trusted third party (Think CA/Trusted Roots in PKI)

   Endorsement is done by chain of signatures. For example the following request ticket is endorsed by two endorser $E1$ and $E2$:

$$\{\{\text{RT}, K_R, Feedback_{E1}\}_{K'_{E1}}, E1, Feedback_{E2}\}_{K'_{E2}}$$

5) **Data Access Ticket (DAT):** is a capability that gives authorization to access a data object. The ticket is issued by the data owner to grant the specified data requester access to the data source and contains the identity of the entity who will access the data. But it is not sent directly to the data source. Additionally it contains an identity to receive the acknowledgment when the ticket is being used.

$$DAT = \{K_O, DOT, \{\textit{Data ID}, \textit{Query}, K_O, K_O^3, K_R\}_{K'_O}\}_{K_S}$$

   In order to protect the real identity of data owners, a data owner may choose to use multiple identities during the ticket exchange process:
   - $K_O$ is the identity that is shared with the data source
   - $K_O^2$ is the identity to be used to communicate with the data requester. This identity is not known by the data source
   - $K_O^3$ is the identity to receive the acknowledgment of ticket being used. This identity is not known by the data requester

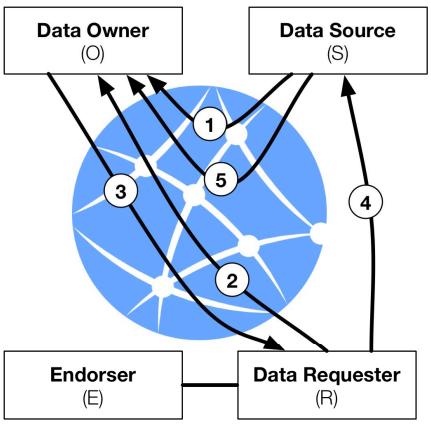


Figure 2. Data Management Protocol with five class of messages transmitted among four roles. Numbers on connections correspond to the message types.

### C. Authorization Messages in the Protocol

In this section, we describe the messages transmitted in the protocol. These five class of messages are illustrated in Figure 2 and a summary is provided in Figure 3. Messages contained in the protocol are:

- **Message 1: Data Source Ticket Generation** The goal of this message is to let the data owner ($O$) know about the existence of a new data object which belongs to the recipient. Furthermore, the included data object ticket allows for the owner to delegate discretionary access to the data object.
  The other included token $\{K_S\}_{K'_O}$ (from $A1$ assumption) authenticates the data source to the data owner. Later, *DOT* can be used to authenticate the recipient.
- **Message 2: Data Request** The data requester broadcasts a data request to the system subscribers. This data request may contain endorsements from one or more endorsers which may provide supplementary information regarding the data request.
  The data owner's client will process the data requests against its library of data object tickets in order to check if participation condition is met, and identify which tickets may meet the specifications of the data query. Endorsements as well as supplementary information can also help the data owner to decide on whether to participate in the data request. Data owner can verify originate of endorsement by checking endorser's identities (from $A3$ endorsement) as well as confirming the $K_R$ in $RT$ is matched with the sender of request.
- **Message 3: Ticket Exchange** When the data owner consents to grant access to the data object(s), a ticket exchange message is sent to the data requester. This ticket contains one or more data access tickets per data source, each one containing part of requested data. Each data access ticket includes identity $K_S$ of the data source which is storing the data object, as well as the address that the data can be accessed by ($DAP$).

$(M1)\ S \rightarrow O : \{DOT\{K_S\}_{K'_O}\}_{K_O}$
$DOT = \{Data\ ID, K_O, metadata, DAP\}_{K'_S}$

$(M2)\ R \rightarrow * : \{\{RT,\ Feedback\}_{K'_{E1}}, K_{E1}, K_R\}_{K'_R}, K_R$
$RT = \{Request\ ID, Query, Conditions,$
$Metadata, Duration, K_R\}_{K'_R}$

$(M3)\ O\ via\ K^2_O \rightarrow R : \{Request\ ID, K^2_O,$
$[\{Data\ ID, DAP, DAT, K_S\}_{K^{2'}_O}]^+\}_{K_R}$
$DAT = \{K_O, DOT, \{Data\ ID,\ Request\ ID,\ Query,$
$K_O, K^3_O, K_R\}_{K'_O}\}_{K_S}$

$(M4)\ R \rightarrow S : \{[\{DAT\}_{K'_R}]^+, K_R\}_{K_S}$

$(M5)\ S \rightarrow O : \{K_S, DOT,$
$\{Query, K_O, K^3_O, K_R\}_{K'_O}\}_{K^3_O}$

Figure 3. Five access control messages in the Data Management Protocol.

- **Message 4: Data Access** As as a result of the message 3, the data requester has now received the data access path as well as any other relevant information needed to access the data. Depending on the application, the data requester may contact the data source(s) directly or indirectly through the system.
  Data source can verify the access by testing:
  1) $DOT$ is signed by the data source $S$ and includes $K_O$. This will verify $O$ right to grant a capability on *Data ID*:
  $DOT \Rightarrow O$ *controls Data ID*
  2) $DAT$ includes signed *Data ID*, *Query*, and $K_R$ with $K_O$ which matched with requester key. This verifies $O$ grant a capability on *Data ID* using *Query* for $R$:

$$\{K_R,\ Data\ ID\ ,\ Query\}_{K'_O} \Rightarrow$$
$$O\ says\ (K_R \wedge\ Data\ ID\ \wedge\ Query)$$
$$(R\ for\ O)\ says\ DataID \wedge Query$$

  3) Optionally, the data source can check *Request ID* against a data request black list.
- **Message 5: Access Announcement** In step 5, it is expected that Data Sources will announce to the system that access has been made utilizing the DAP(s) and credentials provided in Step 3 and accessed as part of Step 4. This allows the data owner to monitor accesses to their data and inform other parties about successful transmission of transaction.

## V. Data Store System

In this section we explain the design of the access control data store system. This storage system supports the distributed access control and is separated from the source data. Separation of source data grants the possibility of keeping source data at its origin. The access control data store component is a persistent, scalable, decentralized, and distributed storage system based on Blockchain. We model the Blockchain as a form of persistent data storage with update notification support. It overcomes the single point of trust and failure, and provides full transparency of transactions very similar to Bitcoin. It has to be noted that the Bitcoin's Blockchain is just an example of an appropriate protocol, possibly, there exists other such appropriate protocols. Before explaining the system, it is necessary to review core components of the system which are adopted from the Bitcoin's Blockchain. Then we show how Blockchain can be used as data storage:

### A. Blockchain

Bitcoin, the system first introduced by Nakamoto [9], is the first truly decentralized global currency system. Like any other currency, its main purpose is to facilitate the exchange of goods and services by offering a commonly accepted value. Unlike traditional currencies however, Bitcoin does not rely on a centralized authority to issue, control the supply, distribution and verification of the validity of transactions. Bitcoin enables a network of computers to maintain a collective bookkeeping via the internet. This bookkeeping is neither closed nor in control of one party. Rather, it is public and available in one digital ledger called Blockchain which is fully distributed across the network and relies on a network of volunteers that collectively implement a replicated ledger and verify transactions. Traditionally, Blockchain has been discussed in the context of Bitcoin, however Blockchain goes beyond the scope of consensus currency, introduces many new and innovative methods for propagating information in the network, public transaction history, multi granularity and many others.

Blockchain uses a multi-hop broadcast to propagate transactions and blocks through the network to update the ledger replicas. In the Blockchain, all transactions are logged including information on the date, time, participants and amount of every single transaction. Each node in the network owns a full copy of the Blockchain and on the basis of cryptographic principles, the transactions are verified by the so-called Bitcoin Miners, who maintain the ledger. The systematic eventual consistency principles also ensure that these nodes automatically and continuously agree about the current state of the ledger and every transaction in it. If anyone attempts to corrupt a transaction, the nodes will not arrive at a consensus and hence will refuse to incorporate the transaction in the Blockchain. So every transaction is public and thousands of nodes unanimously agree that a transaction has occurred on particular date and time. In Blockchain, trust comes from the fact that everyone has access to a shared single source of truth.

In our solution, we use the Blockchain to store the access control data in a decentralized manner. Prior to describing the decentralized access control, we need to discuss the Blockchain as data storage.

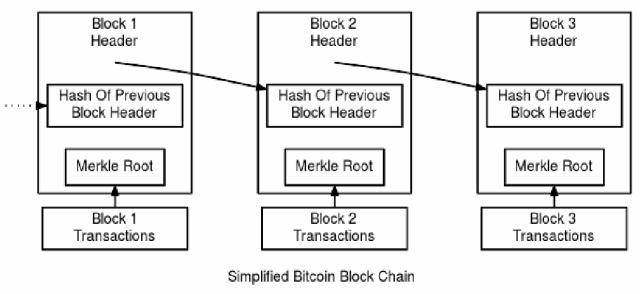


Figure 4. Overview of the Blockchain. Every block contains a hash of the previous block. New transactions are constantly added to the end of the chain, source [9].

### B. Data Model in Blockchain

A block chain is a transaction database shared by all nodes participating in a system. As pointed out earlier, Blockchain can be used as data storage for many different applications; it provides various storage functionalities, among those three primitives are essential to our system: *i.* retrieve, *ii.* update, and *iii.* add. Further, we explain how these primates are supported in the Blockchain and form the data flow, but first we review some definitions in the Blockchain adapted from Nakamoto paper [9] and the Bitcoin Developer Guide [10], illustrated in Figure 4:

- **Transaction:** a transaction is a transfer of value (e.g. Bitcoin, information) that is broadcast to the network and collected into blocks. A transaction references previous transaction outputs as new transaction inputs and dedicates all input Bitcoin values to new outputs. It is possible to browse and view every transaction ever collected into a block.
- **Block:** transaction data is permanently recorded in files called blocks. Blocks are organized into a linear sequence over time (also known as the block chain). As Figure 4 illustrates, every block contains a hash of the previous block. New transactions are constantly being processes by miners into new blocks which are added to the end of the chain and can never be changed or removed once accepted by the network.
- **Mining:** is a distributed consensus system that is used to confirm transactions and add transaction records to the public ledger of past transactions (Blockchain). It enforces a chronological order in the Blockchain and allows different computers to agree on the state of the system.
- **Miner:** is an individual or an organization performing the mining. Miners dedicate considerable computation power for maintaining the Blockchain. In Bitcoin miners are incentivized by a reward i.e. Bitcoins. In our system miners are researchers and organizations requesting and analyzing the data; it is in their benefit to mine the Blockchain.
- **Blockchain:** is a transaction database shared by all nodes participating in a system. A full copy of a Blockchain contains every transaction in order, dating back to the very first one. The entire Blockchain can be downloaded and openly reviewed by anyone.
- **Genesis block:** is the first block of a Blockchain. The genesis block is hardcoded into the software and is a special case in that it does not reference a previous block.

Our system's data flow is based on three essential primitives enabled by the Blockchain:

1) **Retrieve:** per design, the entire Blockchain can be retrieved and downloaded by anyone. With this information, one can find out how much value (e.g. Bitcoin, information) belonged to each entity at any point in history.
2) **Update:** transactions and mining results are broadcasted in the network, every new block is ordered and linked to the previous block which makes it impossible for nodes to miss any added information.
3) **Add:** adding data is the same process as transferring data:
   a) When a node in a Blockchain wants to transfer data, the node broadcasts the request, the request is received by all the nodes on the Blockchain network.
   b) After receiving the request, nodes which are miners will add this most recent transaction request into a block. Then they run the new block and the previous block into a set of hash function based calculations.
   c) All the miners start racing on the complicated cryptographic puzzle. When the first miner solved the block, it adds the block to the end of Blockchain and will broadcast it to its peers.
   d) After the broadcast, peers will check the transaction and will start using the new version of Blockchain.

In the next section, we discuss different mechanisms to allow our clients to connect to the data store system. We use the above primitives and implement a publish-subscribe messaging servie which delivers transactions to their intended recipients.

## VI. Messaging Service

In the world of empowered IoT users, interaction with the data flow is a major issue and it is crucial to adopt the most scalable solution. In general, there are three different client data access models available for users to access and communicate with the data store system: *i.* direct access, *ii.* server-client model, and *iii.* publish-subscribe model.

In this section, we review these three options in the context of the IoT world. We explain how publish-subscribe architecture provides scalable, flexible, and reliable communication between many senders and receivers without the need of persistent connectivity. Furthermore, we explain how we adapt the publish-subscribe model into our solution. In our system, whenever two roles want to interact, one sends

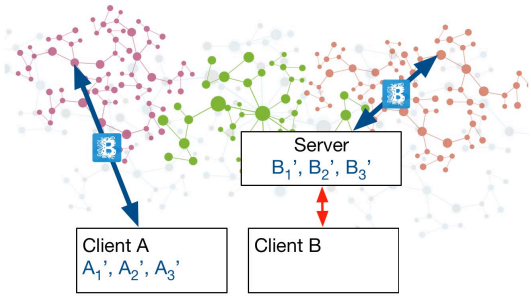


Figure 5. Client Access Models to the BlockChain. Left: Direct Access, Right: Server Client. Blue lines indicate Blockchain access, red line indicates API call.

a message to the messaging service component. Then, the elements of messaging service will send the message to the data store system which is based on Blockchain. After successful storage of the message in the Blockchain (data store system), it will be fetched by the recipient through the messaging service.

### A. Data Access Model

In order to address a variety of use cases we present three different client data access models for data owners and other participants to interact with the Data Flow.

- **Direct Access:** Blockchain openness allows anyone to download the whole distributed database up to the present and subsequent updates as deltas (blocks). (Figure 5-Left) This method is very straightforward and requires minimal implementation. It provides full validation of the Blockchain, as well as maintaining the highest level of safety and privacy. However, this model is not feasible to implement in all environments. This model requires a powerful, always-on computer to handle a large amount of data and to query this data. Devices such as mobile phones, embedded systems or other less powerful systems typically lack the resources to interact directly with the data flow. In reality, each entity whether it is Data Owner, Data Source, or Data Requester, is interested only in a small fraction of data. Excess amount of redundant processing, as each client will need to process and download the entire Blockchain, results in a high amount of waste.
- **Server-Client:** In environments where there is a high degree of trust between two entities a client-server model avoids the redundant processing associated with the Direct Access Model. In a client-server model the server handles all of the client's interactions with the data flow (Figure 5-Right). In these environments the client's public and private keys are transmitted to the server which acts on the client's behalf. Typically the client's interaction with the data flow will be completely abstracted by an application that interprets communication received from the server. Some strengths of this model include: it allows for password resets or other ways for the server to validate the client's identity, in case the client loses its keys. This model introduces risks into the system. Since client's keys are stored on the server, in a server-client environment, if a server is compromised, a client's keys might also get stolen. Since these servers typically store large number of keys, they are often the target of attackers.
- **Publish-Subscribe (Pub-Sub):** This model represents a mixture of the previous two systems. In a Pub-Sub system the Publisher or Server monitors the data flow on behalf of the Subscriber (Figure 6). This substantially minimizes the processing load placed on the data owner. Clients subscribe to a set of queries. The publisher then filters out incoming traffic based on these queries, and only communicates matching queries to the subscriber. In these systems, the publisher does not have access to the data. It communicates the encrypted information, via a subscription, to the subscriber who is then able to decrypt the information with their private key. The traffic will be decrypted and processed on the client side. The only query required in the protocol is matching the destination to an identity. This query can be implemented very efficiently using Bloom filter and provided by the pub-sub model.

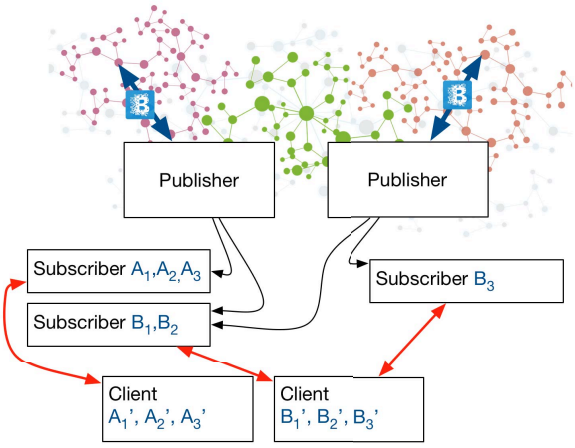


Figure 6. Publish-Subscribe client access model to the Blockchain. Blue lines indicate the Blockchain access, red lines indicate API call, and black lines indicate publish-subscribe data transfer.

Next, we explain how we adopt the publish-subscribe model into our solution.

### B. Publish-Subscribe Model

In this section, we explain how users' clients utilize the subscribers to receive specific updates from the Blockchain. As explained in the previous section, a Blockchain is a collection of blocks. Each block contains a set of transactions such as ticket request and data access, between different roles in the system. By using a publish-subscribe model, publishers join the Blockchain, collect and filter appropriate transactions, and provide subscribers with their requested transactions. This process includes the following mechanisms:

1) **Join:** after joining the Blockchain, each publisher receives updates and will be able to access previous blocks in the Blockchain. Then, subscribers subscribe to one or more publishers, request a dedicated cache space to be initiated on the publisher, and provide them with a set of identities (public keys). The set of identities determines which transactions the subscriber is interested to receive. The cache space is intended to store data so data can be served faster when the subscriber requests it. It should be noted that publishers are located on machines that have access to the Blockchain updates, and also have stored copy of the Blockchain for fast local access. Subscribers are located on the client's machine.
2) **Receive updates:** whenever a new subscriber is added, it is typically specified from which past block it would like to start receiving new blocks. Because of this, publishers can ensure that they are sending updates beginning with the correct block. Ideally all subscribers will receive updates from the same block, i.e. the last added block. Hence, publishers use one reader to retrieve updates, and then relay them to all interested subscribers. In the case of error conditions, the publisher needs to restart sending updates from the last acknowledged location. For this, the publisher may need to read an older update. If many subscribers are recovering simultaneously, a naive implementation, a publisher would read from many positions in blocks simultaneously, which might cause high publisher I/O load. Another approach is to have one recovery thread from the oldest position working towards the end.
3) **Filter:** after receiving the updates from the Blockchain, publisher breaks down the blocks into original transactions. Then, transactions are filtered on the publisher side; the subscribers inform the publishers of what filters they need. The Publisher only delivers updates that meet the specifications of the supplied filters. While the evaluation of filters does place some additional processing overhead on the publisher, it helps conserve both memory and network bandwidth. This is especially the case when there are many subscribers that require only a small subset of the data.
   Subscribers request filtering for a set of identities that may be matched with either the sender or recipient of a transaction. For example, a subscriber may request that a publisher filters messages only set as broadcast to all members. Using this structure we can easily implement a Bloom filter, which works efficiently under these conditions. If a transaction is matched, it will be kept and stored in the cache corresponding to the subscriber.
4) **Deliver:** reliability is an important requirement. For example, one missed update could lead to permanent corruption in a user's portfolio. Subscribers receive the update stream of blocks. Publishers periodically track the delivery of updates by having subscribers acknowledging the delivery of updates. It is assumed that Subscribers are stateless, i.e it is not required for them to keep track of the state of deliveries. When a subscriber asks for receiving its updates, publisher sends all the transactions in the cache corresponding to the subscriber and asks for the subscriber's acknowledgment. Upon receiving the acknowledgment, publisher clears the cache. Otherwise, publisher keeps trying upon receiving the acknowledgment.

It should be noted that one client may deploy many subscribers, and one subscriber could join many unique publishers. Each publisher joins only one Blockchain.

## VII. Discussion and Future Works

Scalability and security correctness are two important requirements for a practical IoT. Intuitively, in our proposed system, these two factors are met through the underlying mechanisms. System scales as the underlying storage layer is based on the Blockchain and access layer is based on the publish-subscribe best practices, both very well studied and practically proven methods. Even though the intuition is sound, further work is necessary to show the scalability with thorough experiments. For the prototype, we plan to use Etherium Blockchain [11]. As for metrics, Blockchain throughput and complexity, publish-subscribe throughput and performance are examples of necessary metrics to evaluate and analyze.

Another important requirement for a practical IoT, is the security correctness of the control layer. Our proposed system is based on delegation by certificate method. This method and its security correctness is described in Abadi et al's [7] and [8]. Further investigation is required to formally prove our proposed capability based access control.

We also realize that sensory power consumption is a major concern in IoT and Blockchain based systems. We believe that intelligent data subscription of sensors is a potential solution. However, further investigation is planned in the future work.

## VIII. Related Works

Other user-oriented and privacy preserving designs in IoT have been proposed [12], [13], [14], [15]. Some models have utilized publish-subscribe methods to support the flexibility in configuring and building assemblies of services or peer-to-peer data management [16], [17], [18]. Others have used capability based security and Access Control Lists, in distributed contexts [19], [20], [21], [22], [23]. For data sharing and privacy-preserving distributed data mining problem, some cryptography solutions have been proposed [24], [25], [26]. However, these methods are commonly impractical at the scale of IoT with billions of things and millions of users.

## IX. Conclusion

This paper describes a new user-oriented world of IoT. In this world, users are empowered by their ability to control the access to the data which knowingly or unknowingly was generated and belongs to them. This data can be requested by

other users and organizations to be collectively analyzed and potentially deliver value to the society. Our solution is based on well-tested principles and practices used in large-scale distributed systems and cryptocurrencies, exploiting capabilities, access lists, publish-subscrib, and Blockchain methods. The scheme has many use cases including healthcare, smart cities, and automated vehicles.

## ACKNOWLEDGMENTS

This research program is supported by a collaborative award from the National Science Foundation (NSF award numbers CNS-1329686, CNS-1329737, CNS-1330142, and CNS-1330491).

## REFERENCES

[1] Z. D. Stephens, S. Y. Lee, F. Faghri, R. H. Campbell, C. Zhai, M. J. Efron, R. Iyer, M. C. Schatz, S. Sinha, and G. E. Robinson, "Big data: astronomical or genomical?" *PLoS Biol*, vol. 13, no. 7, p. e1002195, 2015.

[2] S. Hu, H. Liu, L. Su, H. Wang, T. F. Abdelzaher, P. Hui, W. Zheng, Z. Xie, and J. A. Stankovic, "Towards automatic phone-to-phone communication for vehicular networking applications," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 1752–1760.

[3] Wired. (2016, Jan.) Gm and lyft are building a network of self-driving cars. [Online]. Available: http://goo.gl/PVvDvP

[4] Reuters. (2015, Jan.) Surprise! with $60 million genentech deal, 23andme has a business plan. [Online]. Available: http://goo.gl/AMqJYm

[5] Forbes. (2014, Dec.) Google faces $18 million fine for web privacy violations: Dutch watchdog. [Online]. Available: http://goo.gl/0xq3O5

[6] B. C. Neuman and T. Ts' O, "Kerberos: An authentication service for computer networks," *Communications Magazine, IEEE*, vol. 32, no. 9, pp. 33–38, 1994.

[7] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin, "A calculus for access control in distributed systems," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 15, no. 4, pp. 706–734, 1993.

[8] B. Lampson, M. Abadi, M. Burrows, and E. Wobber, "Authentication in distributed systems: Theory and practice," *ACM Transactions on Computer Systems (TOCS)*, vol. 10, no. 4, pp. 265–310, 1992.

[9] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Consulted*, vol. 1, no. 2012, p. 28, 2008.

[10] Bitcoin developer guide. [Online]. Available: https://bitcoin.org/en/developer-guide

[11] V. Buterin, "A next-generation smart contract and decentralized application platform," *White Paper*, 2014.

[12] J. Al-Muhtadi, R. Campbell, A. Kapadia, M. D. Mickunas, and S. Yi, "Routing through the mist: privacy preserving communication in ubiquitous computing environments," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*. IEEE, 2002, pp. 74–83.

[13] M. Román, C. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt, "A middleware infrastructure for active spaces," *IEEE pervasive computing*, vol. 1, no. 4, pp. 74–83, 2002.

[14] R. Campbell, J. Al-Muhtadi, P. Naldurg, G. Sampemane, and M. D. Mickunas, "Towards security and privacy for pervasive computing," in *Software SecurityTheories and Systems*. Springer, 2003, pp. 1–15.

[15] X. Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher, "Enabling reputation and trust in privacy-preserving mobile sensing," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 12, pp. 2777–2790, 2014.

[16] M. Blackstock, N. Kaviani, R. Lea, and A. Friday, "Magic broker 2: An open and extensible platform for the internet of things," in *Internet of Things (IOT), 2010*. IEEE, 2010, pp. 1–8.

[17] L. Roalter, M. Kranz, and A. Möller, "A middleware for intelligent environments and the internet of things," in *Ubiquitous Intelligence and Computing*. Springer, 2010, pp. 267–281.

[18] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, "Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services," *Services Computing, IEEE Transactions on*, vol. 3, no. 3, pp. 223–235, 2010.

[19] J. Li and A. H. Karp, "Access control for the services oriented architecture," in *Proceedings of the 2007 ACM workshop on Secure web services*. ACM, 2007, pp. 9–17.

[20] G. Sampemane, P. Naldurg, and R. H. Campbell, "Access control for active spaces," in *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*. IEEE, 2002, pp. 343–352.

[21] S. Gusmeroli, S. Piccione, and D. Rotondi, "A capability-based security approach to manage access control in the internet of things," *Mathematical and Computer Modelling*, vol. 58, no. 5, pp. 1189–1205, 2013.

[22] J. Liu, Y. Xiao, and C. P. Chen, "Authentication and access control in the internet of things," in *2012 32nd International Conference on Distributed Computing Systems Workshops*. IEEE, 2012, pp. 588–592.

[23] J. L. Hernández-Ramos, A. J. Jara, L. Marın, and A. F. Skarmeta, "Distributed capability-based access control for the internet of things," *Journal of Internet Services and Information Security (JISIS)*, vol. 3, no. 3/4, pp. 1–16, 2013.

[24] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[25] V. Oleshchuk, "Internet of things and privacy preserving technologies," in *2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace&Electronic Systems Technology*, 2009.

[26] C. C. Aggarwal and S. Y. Philip, *A general survey of privacy-preserving data mining models and algorithms*. Springer, 2008.