Building a secure infrastructure for IoT systems in distributed environments

Xiaoyang Zhu

N°d'ordre NNT : 2019LYSEI038

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
INSA-Lyon

Ecole Doctorale N° 512
INFOMATHS

Spécialité/discipline de doctorat :
Informatique

Soutenue publiquement le 24/06/2019, par :
Xiaoyang Zhu

Building A Secure Infrastructure for IoT Systems in Distributed Environments

---

Devant le jury composé de :

POTOP-BUTUCARU, Maria	Professeure des Universités, UPMC	Rapporteure
DRIRA, Khalil	Directeur de Recherche, Laboratoire CNRS LAAS	Rapporteur
MORIN, Christine	Directrice de Recherche, INRIA Rennes	Examinatrice
BRUNIE, Lionel	Professeur des Universités, INSA-Lyon	Examinateur
BADR, Youakim	Maître de Conférences HDR, INSA-Lyon	Directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **http://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage **secretariat@edchimie-lyon.fr** INSA : R. GOURDON	M. Stéphane DANIELE Institut de recherches sur la catalyse et l'environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 Avenue Albert EINSTEIN 69 626 Villeurbanne CEDEX **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **http://edeea.ec-lyon.fr** Sec. : M.C. HAVGOUDOUKIAN **ecole-doctorale.eea@ec-lyon.fr**	M. Gérard SCORLETTI École Centrale de Lyon 36 Avenue Guy DE COLLONGUE 69 134 Écully Tél : 04.72.18.60.97 Fax 04.78.43.37.17 **gerard.scorletti@ec-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : H. CHARLES **secretariat.e2m2@univ-lyon1.fr**	M. Philippe NORMAND UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX **philippe.normand@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://www.ediss-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : M. LAGARDE **secretariat.ediss@univ-lyon1.fr**	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 Avenue Jean CAPELLE INSA de Lyon 69 621 Villeurbanne Tél : 04.72.68.49.09 Fax : 04.72.68.49.16 **emmanuelle.canet@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 **infomaths@univ-lyon1.fr**	M. Luca ZAMBONI Bât. Braconnier 43 Boulevard du 11 novembre 1918 69 622 Villeurbanne CEDEX Tél : 04.26.23.45.52 **zamboni@maths.univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction **ed.materiaux@insa-lyon.fr**	M. Jean-Yves BUFFIÈRE INSA de Lyon MATEIS - Bât. Saint-Exupéry 7 Avenue Jean CAPELLE 69 621 Villeurbanne CEDEX Tél : 04.72.43.71.70 Fax : 04.72.43.85.28 **jean-yves.buffiere@insa-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction **mega@insa-lyon.fr**	M. Jocelyn BONJOUR INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69 621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **http://ed483.univ-lyon2.fr** Sec. : Véronique GUICHARD INSA : J.Y. TOUSSAINT Tél : 04.78.69.72.76 **veronique.cervantes@univ-lyon2.fr**	M. Christian MONTES Université Lyon 2 86 Rue Pasteur 69 365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Abstract

The premise of the Internet of Things (IoT) is to interconnect not only sensors, mobile devices, and computers but also individuals, homes, smart buildings, and cities, as well as electrical grids, automobiles, and airplanes, to mention a few. However, realizing the extensive connectivity of IoT while ensuring user security and privacy still remains a challenge. There are many unconventional characteristics in IoT systems such as scalability, heterogeneity, mobility, and limited resources, which render existing Internet security solutions inadequate to IoT-based systems. Besides, the IoT advocates for peer-to-peer networks where users as owners intend to set security policies to control their devices or services instead of relying on some centralized third parties. By focusing on scientific challenges related to the IoT unconventional characteristics and user-centric security, we propose an IoT secure infrastructure enabled by the blockchain technology and driven by trustless peer-to-peer networks. Our IoT secure infrastructure allows not only the identification of individuals and collectives (e.g., companies, families, organizations) but also the trusted identification of IoT things (e.g., devices, services) through their owners by referring to the blockchain in trustless peer-to-peer networks. The blockchain provides our IoT secure infrastructure with a trustless, immutable and public ledger that records individuals and collectives identities, which facilitates the design of the simplified authentication protocol for IoT without relying on third-party identity providers. Besides, our IoT secure infrastructure adopts socialized IoT paradigm which allows all IoT entities (i.e., individuals, collectives, things) to establish relationships and makes the IoT extensible and ubiquitous networks where owners can take advantage of relationships to set access policies for their devices or services. Furthermore, in order to protect operations of our IoT secure infrastructure against security threats, we also introduce an autonomic threat detection mechanism as the complementary of our access control framework, which can continuously monitor anomaly behavior of device or service operations. At last, we prototype our solution, present use cases, and run experiment and simulation which show that our proposed IoT secure infrastructure can effectively interconnect all IoT entities through our authentication and authorization mechanisms and detect both known and unknown threats with high detection rates and low false positive alarms.

Keywords – Internet of things, Blockchain; Identity; Authentication; Authorization; Threat detection.

Résumé

Le principe de l'Internet des objets (IdO) est d'interconnecter non seulement les capteurs, les appareils mobiles et les ordinateurs, mais aussi les particuliers, les maisons, les bâtiments intelligents et les villes, ainsi que les réseaux électriques, les automobiles et les avions, pour n'en citer que quelques-uns. Toutefois, la réalisation de la connectivité étendue de l'IdO tout en assurant la sécurité et la confidentialité des utilisateurs reste un défi. Les systèmes IdO présentent de nombreuses caractéristiques non conventionnelles, telles que l'évolutivité, l'hétérogénéité, la mobilité et les ressources limitées, qui rendent les solutions de sécurité Internet existantes inadaptées aux systèmes basés sur IdO. En outre, l'IdO préconise des réseaux peer-to-peer où les utilisateurs, en tant que propriétaires, ont l'intention d'établir des politiques de sécurité pour contrôler leurs dispositifs ou services au lieu de s'appuyer sur des tiers centralisés. En nous concentrant sur les défis scientifiques liés aux caractéristiques non conventionnelles de l'IdO et à la sécurité centrée sur l'utilisateur, nous proposons une infrastructure sécurisée de l'IdO rendue possible par la technologie de la blockchain et pilotée par des réseaux peer-to-peer sans confiance. Notre infrastructure sécurisée pour l'IdO permet non seulement l'identification des individus et des collectifs (entreprises, familles, organisations), mais aussi l'identification fiable des objets de l'IdO (dispositifs, services) par leurs propriétaires en se référant à la blockchain dans les réseaux peer-to-peer sans confiance. La blockchain fournit à notre infrastructure sécurisée de l'IdO une base de données fiable, immuable et publique qui enregistre les identités individuelles et collectives, ce qui facilite la conception du protocole d'authentification simplifié de l'IdO sans dépendre des fournisseurs d'identité tiers. En outre, notre infrastructure sécurisée pour l'IdO adopte un paradigme d'IdO socialisé qui permet à toutes les entités de l'IdO (les individus, les collectifs, les choses) d'établir des relations et rend l'IdO extensible et omniprésent les réseaux où les propriétaires peuvent profiter des relations pour définir des politiques d'accès pour leurs appareils ou services. En outre, afin de protéger les opérations de notre infrastructure sécurisée pour l'IdO contre menaces de sécurité, nous introduisons également un mécanisme autonome de détection des menaces en complément de notre cadre de contrôle d'accès, qui peut surveiller en permanence le comportement anormal des opérations de dispositif ou de service. Enfin, nous réalisons le prototype de notre solution, nous présentons des cas d'utilisation et nous effectuons des expériences et des simulations qui montrent que l'infrastructure sécurisée que nous proposons est capable d'interconnecter efficacement toutes les entités de l'IdO grâce à nos mécanismes d'authentification et d'autorisation et de détecter les menaces connues et inconnues avec des taux de détection élevés et de faibles alarmes positives erronées.

Mot-clefs – Internet des objets; Blockchain; Identité; Authentification; Autorisation; Détection de la menace.

To my family.

Acknowledgements

I would like to thank Dr. Youakim Badr for his guidance during my Ph.D. He has always been kind and open-minded, which allows me to do the research freely. I am also particularly grateful to Dr. Salim Hariri, Dr. Jesus Pacheco from University of Arizona, and Dr. Victor Hugo Benitez-Baltazar from University of Sonora for the collaborations, which, as the supplementary, significantly improve the completeness of our solutions. I also want to express my gratitude to Dr. Wanli Su who guided me into the field of information security during my undergraduate studies, and Dr. Hui Li for his guidance during my graduate years, which laid a solid foundation for my Ph.D. research.

Besides, I would like to thank my family and friends. First and foremost, I am most grateful to my parents who continuously support and encourage me as always. They have provided me with a very stable family condition for so many years, which allows me to find and pursue my own life as I dreamed. Moreover, I am glad to see my young sister graduate from her college, wish her a brighter future, and live the life she wants. Additionally, I want to express my gratitude to my colleagues, especially Dr. Arthur Gatouillat for their exceptional care and help in my daily life at INSA de Lyon.

Finally, I would like to thank the China Scholarship Council (CSC) and the Chinese government for the funding so that I can study in France. Understanding that it is not easy for a country to cultivate a Doctor, I will return to China and make my contribution to the prosperity and rejuvenation of our country.

List of Figures

List of Tables

Table of Contents

CHAPTER 1

Introduction

Contents

1.1 IoT Background and Security

The Internet of things (IoT) is a global infrastructure for the information society, in which connected objects, intelligent systems and software applications gather data from the physical world, process information and offer services to users [ITU12b]. The premise of IoT is to allow anyone to access anything through any device from anywhere at any time. In the past decades, the IoT has emerged, grown and gradually affected the daily lives of human beings in many new application domains, ranging from wearable devices, smart manufacturing, smart gird, to smart homes and ambient intelligence, to mention a few. According to the report published by Gartner [Gar17], 8.4 billion IoT units were deployed in 2017, and there will be over 20 billion connected devices that fall under the rubric of the "Internet of Things" in 2020. However, the security in IoT has not received sufficient attention proportional to its rapid growth and applications. The SANS Institute has initiated a survey [SAN14] on the IoT cybersecurity in 2014. The respondents came from an extensive range of industries and had different roles in companies which had different proportions of a domestic and international workforce. At that time, only 17.2% of the respondents thought that IoT security

could be a disaster and more severe than security problems on the Internet. In other words, only a few paid highly close attention to IoT security.

- In December 2014, attackers infiltrated a Germany steel mill facility and obtained control right of the cooperate and plant network, which leads to the explosion of a furnace [Cob15].
- The recall event of 1.4 million vehicles [GG15] in 2015 due to security breaches of the controller area network (CAN) denotes that the IoT security problems could be disastrous and fatal.
- In October 2016, the Mirai [Cor16] IoT botnet infected numerous IoT devices and initiated distributed Denial of Service (DoS) attacks through flooding DNS servers, which results in the large-scale Internet network paralysis.
- The medical device hijack (MEDJACK) raises great attention in the MEDJACK.3 special session of the RSA 2017 [RSA17]. Medjack ransomwares have blackmailed much money from hospitals and medical device manufacturers since 2015.

With the popularity of IoT, billions of devices have been inter-connected into IoT systems, which significantly increase the attack surface and possibilities for hackers to get unauthorized access and undermine these systems. Without appropriate security solutions, various IoT systems will never be deployed globally due to all kinds of security concerns. Therefore, the security in IoT should be then taken into consideration not only at design-time of IoT systems but also at runtime.

1.2 Research Challenges

In this section, we synthesize the research challenges of building a secure infrastructure for IoT systems. Since IoT has the same security issues with the Internet such as WiFi security, 3/4G security and Internet Protocol (IP)-based security, some of the existing security solutions are still suitable for securing IoT systems. However, there are so many particularities in IoT that most of the security solutions, which have been developed for the Internet cannot be applied or adapted to IoT systems [Win15]. The lack of consensus on how to implement security of IoT mainly arises from unconventional characteristics of IoT. Current security solutions [Win15] fail to address these characteristics to mitigate security threats effectively. As a result, we firstly cover the challenges coming from four IoT unconventional characteristics namely scalability, heterogeneity, dynamic changes, and limited resources, and emphasize on their impacts on building IoT security solutions. In addition to challenges coming from unconventional characteristics, IoT advocates for peer-to-peer networks where users as owners intend to set security policies to control their devices or services instead of relying on some centralized third parties. Therefore, the user-centric IoT security calls for a resilient and trustworthy identity management framework and a supporting access control framework with dynamic defense. The former can identify all IoT entities and authenticate legitimate users without relying on third parties in distributed peer-to-peer networks while the latter can grant corresponding permissions to legitimate users in trustless peer-to-peer networks and dynamically prevent mali-

cious users from breaching the system and stealing sensitive information when the security system is running. Consequently, we also point out the challenges from managing digital identities (including authentication mechanisms) and access controls which are considered as the keystone of building a secure infrastructure for IoT systems.

1.2.1 Challenges of IoT Unconventional Characteristics

The basis of making the IoT a reality relies on its ability to connect billions of things (i.e., devices, systems, API, sensors, ...). As a result, scalability and heterogeneity are intrinsic characteristics of IoT systems. Besides, device limited resources and their ability to evolve in dynamic environments are also silent characteristics. These characteristics have an impact on designing IoT systems and defining security requirements.

- **Scalability**. IoT refers to interconnecting everything in the cyber-physical world. Any server architecture in IoT would ideally be highly scalable, and be able to support millions of devices all continually sending, receiving, and acting on devices and their data. Therefore, scalability is essential to characteristics of future IoT-based systems [Fre15]. In such vast networks of interconnected objects, designing IoT related frameworks such as identity management, authentication, and authorization mechanisms should fully consider the scalability. For instance, digital identity management (i.e., identification of a particular object) should be scalable to unify all IoT entities and provide the authentication mechanism to users in the IoT-based systems. In other words, all IoT entities from people, organizations to devices, asset or even services, should be identified and have the proof to be authenticated for granting authenticated and secure access to users from everywhere at any time. Besides, the vast network scale of IoT devices exposes them to threats of physical attacks and tampering as well [P+09; Ren+11]. Consequently, scalability implies that the IoT infrastructure should be scalable and distributed peer-to-peer networks without centralized control of any security authority (i.e., identity providers, central access servers). The decentralized control of security authority pushes the security control to the edge to enable user-centric security.
- **Heterogeneity**. IoT systems should have the ability to connect any things, ranging from entities in the physical world such as individual, companies, organizations, and devices, to things in the virtual world like applications and, computational and storage resources [ITU12a]. Despite the heterogeneity of entities, they should be able to identify and understand each other by, ideally, a kind of a universal protocol. Without the connectivity of heterogeneous things, it would be impossible to build the IoT. However, we could easily observe heterogeneous issues even only from network communication aspects as shown in Table 1.1, which illustrates all kinds of communication protocols according to the communication range from Personal Area Networks (NFC - ULE Alliance), Local Area Networks (WiFi), Metropolitan Area Networks (Dash7 - WiMAX), to Wide Area Networks (GSM - Symphony Link). Therefore, when designing secure infrastructure for IoT

systems, an interoperable solution for heterogeneity is critical to integrate different platforms, network protocols, IoT devices produced by diverse manufacturers and powered by different operating systems and communication protocols (i.e., RFID, Bluetooth, or none IP-based protocols). For example, the cross-domain authentication issues and networks divergence problems (e.g., Ad hoc networks, the Internet, 3/4G networks) call for a decentralized authentication and authorization paradigm for heterogeneous networks instead of a centralized control center to authenticate and grant access permissions which is not realistic for heterogeneous IoT environments and will definitely cause many problems such as single point failure. To sum up, the interoperability solution in such heterogeneous environments could make all IoT entities interconnected, produce compatible data and provide services to each other [Ele14]. Consequently, interoperability for heterogeneity IoT environments is an inevitable building block in designing security solutions for IoT systems.

- **Dynamic changes in IoT systems and environments.** In the context of IoT, states often describe devices' behaviors. Transitions between states are quite common and frequent, e.g., started and standby, sleeping and waking up, leaving and joining networks [ITU12a]. Besides, the number of connected devices can also evolve. Environments in which IoT devices operate are subject to contextual changes. The characteristic of dynamic changes is the intrinsic properties of the IoT. However, many threats emerge due to dynamic changes in IoT systems. For instance, in intelligent transportation systems with characteristics of the high mobility of connected vehicles, rapidly changing network topology and unbounded network size, hackers could even hijack a moving car and take the control [Gre15]. Particularly, IoT devices, such as vehicles or wearable devices equipped with strong mobility, often make great demands on across domain authentication and authorization to prevent malicious attacks from adversaries. Therefore, the secure IoT infrastructure should be able to resilient to this dynamic changes environment and provide a peer-to-peer authentication and authorization services.
- **Limited and constrained resources.** Many devices of IoT have constraints in their computation, storage, bandwidths and power resources. These constraints directly affect security considerations when developing new security solutions against security threats. For example, key management in wireless sensor networks is indispensable in order to secure the communication channel. However, an ordinary sensor in wireless sensor networks may have very few computations and memory resources (e.g., TelosB with 8 MHz CPU, 10kB RAM, and 250 kbps data rate). Therefore, restricted to limited resources, various resource-saving key management schemes such as polynomial-based, matrix-based, tree-based key predistribution schemes, ECC-based asymmetric and identity-based key agreement schemes, are proposed for these IoT sensors [ZV10]. Besides, these connected IoT devices are continually collecting and delivering data which will consume constrained power and computation resources. Consequently, limited resources like communication channels and batteries are easy to be

Table 1.1 – Heterogeneous communication protocols in IoT

	Range	Standards	Features	Application
NFC	< 0.2 m	ISO/IEC 18000-3	Consumes far less power and does not require pairing	Smart Wallets/Cards, Action Tags, Access Control
ANT	30 m	ANT Wireless protocol stack	Similar to Bluetooth low energy, but is oriented towards usage with sensors.	Sensors
BLUETOOTH SIG	1-100m		Low interference, enhanced range, low cost	Hands-free headsets, key dongles, fitness trackers
RFID	10cm to 200m	ISO/IEC 18000-7:2009		Road tolls, Building Access, Inventory
ENOCEAN	300m, 30m		Energy harvesting, ultra-low power management	Wireless switches, sensors, and controls
X10	PAN	X10 power line, RF protocols	Industrial Protocol for Electronic devices	Smart Home (X10 has No encryption)
Z-WAVE	100m	GFSK	Low power sub 1 GHz RF and works within a mesh topology. 1000 kinds of devices	Smart Home
WI-SUN	Wireless PAN	IEEE 802.15.4g Wireless SUN		Gas Metering, Distribution Automation, M2M, WSN
ZIGBEE	75m but can be extended	ZigBee Home Automation 1.2	Reliable, robust, low-power, scalable, secure, global	Smart Energy, Home Automation, Light Link, Remote Control
6LoWPAN	Wireless PAN	Based on IEEE 802.15.4	Combined IPv6 and low power wireless PAN	Thread
ULE Alliance	Wireless PAN	DECT Specification (digital wireless technology)	Low energy and low latency wireless networks	Smart Home
WiFI	100m but can be extended.	Based on IEEE 802.11 Wireless LAN	High power consumption	Routers, Tablets
Dash7	5 km	Based on ISO 18000-7	Ultra-low power mid-range sensor and actuator communication	Building Automation, Smart Energy, Location-based Services, Automotive
Weightless SIG	Up to 10km	Weightless-N/W/P	Ultra-low cost, ultra low power consumption, excellent signal propagation characteristics for IoT	Smart meters, traffic sensors, industrial monitoring
WiMAX	50 km	Based on IEEE 802.16 Wireless MANs	Low cost and flexibility affordable solution for 4G	Smart metering, M2M, In-vehicle applications
GSM/3G/4G/LTE	WAN (Wide Area Network)	Based on IEEE 802.20		
LoRa Alliance	WAN (Wide Area Network)	LoRa FSK	Low power for sensors	M2M, Smart City, Industrial Applications
NB-IoT	WAN (Wide Area Network)	3GPP Narrowband Cellular Standards	Indoor coverage, low cost, long battery life, high connection density	Smart City
Symphony Link	WAN (Wide Area Network)	Built on the IEEE 802.15.4 standard with the LoRa PHY	Carrier-grade connectivity, suitable for worldwide sub-GHz deployment.	

depleted. Some adversarial applications may exploit these constraints to attack and drain batteries. IoT secure infrastructure thus should take into account devices with limited resources to enable security mechanisms. The constrained IoT environments require lightweight authentication and authorization mechanisms to protect services provided by resource-constrained IoT devices.

In summary, when building an IoT secure infrastructure for real-world applications, challenges from these unconventional characteristics, namely, scalability, heterogeneity, dynamic changes, and limited resources in IoT systems and environments should be taken into account comprehensively.

1.2.2 Challenges from IoT Digital Identity Management

Challenges from managing digital identities (including authentication mechanism) are also critical obstacles in order to build a secure infrastructure for IoT systems. The digital identity refers to a set of information (identifier, credentials, and attributes) used for uniquely identifying an entity in a given context [ITU09]. Given billions of people, trillions of IoT devices, and innumerable data resources, the significant challenge of IoT security is to uniquely identify these entities and allocate digital identities to individuals and things in decentralized networks so that IoT entities could be easily identified and communicate with each other. All services in information systems including the Internet of Things (IoT), at its heart, rely on the digital identity concept to build security mechanisms such as authentication and access control. Without digital identities, entities could barely transact with others, leading to untrusted interactions and consequently the lack of business opportunities.

An identity management system (IdMS) refers to the management of identity information through a set of operations like register, update, revoke and look-up. Figure 1.1 depicts a sample instance of traditional identity management systems, which comprises three main stakeholders: subject (a.k.a user), relying party (also called service provider) and identity provider (IdP) [BT11]. The different three parties are interdependent entities: the subject requests access to services from the relying party which requires the IdP to challenge the subject identity through the authentication protocol. However, when building IdMS for the Internet of Things, traditional IdM systems, or even decentralized IdMS model such as OpenID [RR06], are subject to new challenges such as scalable deployment, the efficiency of across-domain authentications and unconditionally trust by relying on IdPs. For instance, centralized or decentralized IoT Identity Management solutions become quickly obsolete due to distributed IoT networks composed of millions or even billions of devices and objects [RR06; HM05; CS05]. Internet of Things is expected to comprise billions of individuals, collectives and everything in the cyber-physical world, which demands highly scalable IdM systems. Applying traditional centralized IdM scheme – where all IoT identities are maintained by one universal third party – to build a centralized IdM solution becomes unrealistic. Inevitably, there will be many different identity providers using different IdM systems. Albeit, federated identity management solutions like SAML [HM05] and Shibboleth [CS05] allow different identities from different IdM systems to inter-

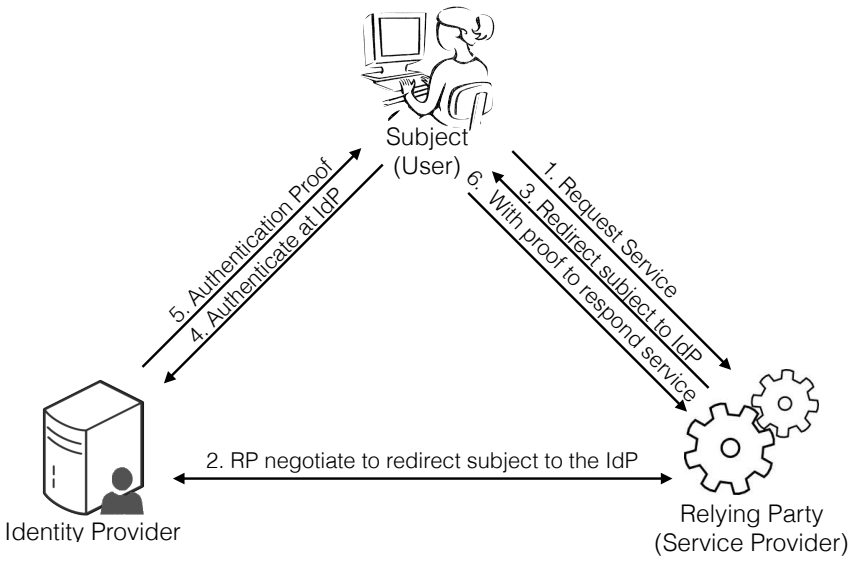


Figure 1.1 – Stakeholders in the traditional IdMS model

operate through the adoption of common standards within the same federation and bring a silver lining of designing the IoT IdMS. However, in trustless networks such as the IoT, the main question is how to build mutual trust relationships between different IdM systems. Consequently, proposed identity management for distributed trustless IoT networks should be scalable with capabilities to establish trust relationships among subjects without centralized control of any security authorities (i.e., identity providers, central access servers, ...).

In addition to the scalability, other IoT unconventional characteristics such as heterogeneity and mobility, also make current IdMS challenging to design and deploy. As demonstrated in the following paragraphs, these characteristics are essential requirements for building scalable and distributed IdMS in the context of the IoT. The spectrum of connected devices makes them extremely heterogeneous with different communication, storage, and processing capabilities. Each device would be subjected to various technologies such as wireless communication technologies (IEEE 802.15.4, WiFi, Bluetooth Low Energy, ...), communication protocols (CoAP, LORA, MQTT, ...), cellular communication technologies (GSM, UMTS, LTE, ...) and hardware-dependent controllers (Arduino, Intel Edison, Raspberry, Eaglebone,...). Diversity and heterogeneity lead to interoperability problems; therefore, unifying all digital identities of IoT devices from different manufacturers, vendors, communities, and standard groups, has been considered to be a mission impossible. As IoT continues to evolve, emerging standards (e.g., OneM2M, IoT reference architecture) remains highly fragmented in terms of vocabularies, methods, and models. The design of interoperable IoT IdM systems remains thus balkanized without an integrated approach to make substantial progress in reducing software, hardware, and communication heterogeneity. The IoT is a ubiquitous environment by which IoT devices, such as vehicles or wearable devices are subject to spatial mobilities when they operate. No matter where these devices are located, subjects need to authenticate themselves, get authorization and access controls to device services. Therefore, IoT

identity management systems should consider mobility and provide peer-to-peer authentication and authorization services.

The design of IoT IdMS is also confronted with challenges from the security perspective. For instance, most of the IdMS solutions are designed under the assumption that subjects and relying parties trust the identity providers. This assumption increases threats from internal attacks of IdMS and hence compromises security services like confidentiality and integrity. Besides, a typical IoT IdMS should be robust enough to defend some longstanding security vulnerabilities or attacks such as single-point failure and phishing [TNP13; AHS11], which undermines the availability of IdMS. Therefore, eliminating trusted third parties and building trusted IdMS in trustless IoT networks are pretty critical issue towards the development of future IoT services. Besides, IdM systems as intermediators are responsible for dealing with all transactions between subjects and relying parties and hence can see all interactions between subjects and relying parties. Consequently, the IoT IdMS should be equipped with capabilities of providing anonymity or pseudonymity to relying parties in order to preserve privacy. Based on our previous discussion, we summarize the following three Identity Challenges (IDC) that hinder the development of scalable and distributed IdM systems for IoT:

IDC-A. How to classify IoT entities and assign IdMS roles to IoT entities?

Entities in the context of IoT involve many aspects including people, things, data, services. Therefore, the key challenge is how to define an extensible and trustful entity structure, representing a specific domain or a security authority, to govern all its entities and assign them corresponding roles in the identity system model. As shown in Figure 1.1, there are three different roles namely, subject, relying party (also called service provider) and IdP in identity management model. In terms of IdM design benefits, an extensible entity structure could solve the identification problem of all its entities. Any new entity could be created and managed within the extensible entity structure, ensuring IdMS scalability. On the other hand, entities assigned to different IdM roles could interact with each other through the universal identity management in IdMS, which facilitates interoperability. Moreover, the appropriate roles assignment also could substantially decrease the risk of phishing attacks from relying parties.

IDC-B. How to build resilient and distributed identity providers?

Trillions of people, innumerable things, massive data volumes, and applications require resilient identity provider systems that ensure scalability and interoperability. They also should provide capabilities of resisting to attacks like phishing and single point of failure. Unlike traditional centralized or decentralized IdP models (i.e., OpenID), a distributed IdP model is appealing and suitable for IoT. However, many critical issues are requiring particular attention such as the identifiers, the efficient management of identity credentials and the organization of identity attributes. These issues need to be handled in the process of designing

IoT distributed IdP to cope with latency, the time delay due to IdP discovery, and availability of identity providers.

IDC-C. How to build trustworthy identity providers in trustless networks?

Privacy is compromised by traditional centralized or decentralized identity providers, where users have to grant full trust to third parties, assumed to be "trusted", in order to obtain digital identities and manage their credentials. Since identity providers manage users meta-data and session logs, user privacy is thus comprised. In this context, it is the assiduous pursuit to replace these "trusted" third parties with more reliable and trusted oracles that could be universally admitted by any users to create their identities by themselves in peer-to-peer networks.

1.2.3 Challenges from IoT Access Controls

Access control refers to a security mechanism which regulates who can access what kind of resources or services in computer systems. The development of information systems since the 1970s has given rise to various access control models in order to handle increasing application requirements. Originated the access control matrix [But74], access control models such as access control lists (ACLs), capability-based access control (CAC), role-based access control (RAC), relationship-based access control (RBAC) and attribute-based access control (ABAC) models, are improved accesses management over users or resources. For instance, the RAC model is designed to manage users efficiently. Through assigning roles to users, administrators can easily manipulate the process of granting access permissions instead of granting permissions to each user. Similarly, the RBAC model is another refined management over users, in which owners of data (e.g., photos on social networks) take advantage of relationships with others to manage access permissions. The ABAC model is born to meet requirements of intensive management over resources in terms of multi-factors such as location, time, battery life.

With the advent of IoT era, many traditional access control models such as the ACLs and RAC models[San+96], which are designed for centralized systems, become obsolete due to the rapid growth of roles and policies. Besides, more and more factors and parameters like temporal and spatial attributes should also be taken into consideration when designing access control solutions. Albeit, the ABAC model aims at handling this problem, the existence of centralized identity providers using the ABAC model still has a scalability issue. A common problem of existing authorization solutions stems from centralized administrative parties (i.e., administrators or identity providers) that become indispensable for assigning access rights, roles, and attributes, and, consequently, these solutions are not suitable for scalable decentralized IoT systems. The CAC model[Her+13; GPR13; HBF17] raises much attention by virtue of its flexibility. However, It has the same premise by which users who request services need to rely on the authentication of third parties like identity providers or certificate authorities. This premise is unsuitable with regard to our vision for trustless IoT environments where users (identities) could be generated by each subject (e.g., human) without the endorse-

ment of other intermediate parties. In other words, the CAC model only works in trusted environments. Therefore, none of the proposed access control models (i.e., ACLs, RAC, ABAC, or CAC) can satisfy the scalable, interoperable, and trustless IoT environments. Although the RBAC model allows owners to set their access control policies without relying on the authentication of third parties, the current RBAC model is restricted to closed online social network platforms such as Facebook, Twitter, or Instagram. Consequently, owners cannot fully control their data or asset in that these operating companies of social networks are the actual controllers of users' data. Besides, these access control models are defined at design time of IoT systems. In practice, protecting IoT systems is a grand challenge at runtime due to the significant increase in the attack surface [Nas+09]. The comprehensive interconnections between IoT devices expose vulnerabilities of IoT systems to attackers. Even devices, which are intended to only operate in local area networks, are sometimes connected to the Internet due to careless configuration or to satisfy particular needs (e.g., they need to be remotely managed). As a result, devices can be easily compromised and become subject to cyber-security risks and attacks with severe impacts (e.g., life-threatening scenarios) [PH16; VRR07]. To sum up, the Access Control Challenges (ACC) are focusing on:

ACC-A. How to design the access control framework which could grant corresponding permissions to legitimate users in trustless peer-to-peer networks?

ACC-B. How to prevent malicious users from breaching the system and stealing sensitive information when the security system is running?

1.3 Security Requirements for IoT Secure Infrastructure

In addition to the previously mentioned challenges, we still have to sort out specific security requirements in order to build our secure infrastructure for IoT. Some researchers have already introduced many works related to security requirements in IoT. For instance, Sicari et al. [Sic+15] not only point out the challenges coming from intrinsic IoT characteristics such as scalability, heterogeneity, and mobility but review security and privacy requirements in terms of confidentiality, authentication, access control, and trust. Moreover, Mahmoud et al. [Mah+15] have identified security challenges in each layer of IoT and potential attacks like replay, DoS, man-in-the-middle, and eavesdropping. Similarly, Vasilomanolakis et al. [Vas+15] present security and privacy requirements from the perspective of identity management, network security, privacy and trust.

From these research papers, we can see, security requirements in IoT should not only cover functional properties of IoT-based systems but consider several layers, ranging from devices, networks, services, and applications layers during the design time and the run-time. On the one hand, the security requirements of IoT secure infrastructure should be under the premise of taking into account unconventional characteristics to meet basic security properties namely, confidentiality, integrity, and availability. On the other hand, an appropriate analysis of security threats at design time and continuous monitoring of vulnerabilities at

runtime, should reduce security breaches and evade potential threats. In other words, the IoT security should thus be considered from bottom-up so that security and privacy protocols should protect the IoT from being undermined by security threats. Referring to the IoT reference model and security capabilities illustrated in [ITU12a], we briefly state main security requirements for a secure IoT infrastructure as follows:

- **Confidentiality** means preventing sensitive data from being retrieved and cracked by unauthorized and malicious parties. Therefore, it should be built on authentication and authorization mechanisms. The confidentiality at the device layer consists of two parts: data storage and data transmission confidentiality. The former one protects the data including programs in devices from disclosure and tampering. The latter is responsible for data confidentiality in communication. In the network layer, it is mainly about the network packages (including signaling data and user data) confidentiality in transmission across networks. The confidentiality in the service layer refers to the confidentiality of data management in the process of storage and computing. In other words, data stored at third-party service providers need confidentiality mechanism to prevent from stealing information by malicious attackers. Otherwise, these centralized service providers such as Facebook [IF18] and Equifax [CSP17] are straightforward to become the honey pot of intruders. Lastly, the confidentiality at the application layer refers to the confidentiality of application data with respect to specific users.
- **Integrity** protects data or programs from being tampered by unauthorized users. At the device layer, integrity not only grantees that data is not altered but ensures that devices and built-in programs are authenticated, trusted, and not hijacked by malicious attackers [BKS15]. Moreover, the integrity of the network layer refers to the integrity protection of user data and signaling data [ITU12a]. The former is often tied to confidentiality protection while the latter helps to avoid DoS attacks in the network layer. Similarly, the integrity protection at the service and application layers also comes from two aspects: users data and programs.
- **Availability** ensures that all IoT services and devices are accessible and resilient to various malicious attacks. It emphasizes the security of IoT systems at runtime in that only running IoT systems can provide services to others. If IoT systems are not deployed, there are even no services provided to users, and hence the availability of services has no meaning. In contrary, other security requirements are more concerned about the correctness of these authentication, authorization, confidentiality and privacy mechanisms when designing these IoT security systems. Therefore, we can say, they are focusing on the IoT security at design time. The systems with high availability could ensure the interconnectivity and accessibility of services provided by these IoT systems while systems with low availability could incur many security concerns such as attacks on the reliability. For instance, attacks targeting the reliability of communications between things, such as capture attacks, impersonate attacks [CC13], could gain control of IoT systems and retrieve relevant information that might influence the runtime security of the entire IoT system. The device layer availability protects devices from

physical attacks and DoS attacks aimed at the limited resources in IoT systems. The availabilities in the network layer, service layer and application layer are very close. They try to prevent DoS attacks and guarantee the availability of the networks, services, and applications.

- **Privacy** is closely related to users. For example, the target of the RFID tag tracking attack and the eavesdropping attack is all about the individuals. Attackers can make use of hidden RFID readers to track insecure tags of products and retrieve sensitive information like location information and even credit card information used to pay these products [Dim05]. The privacy-preserving aims to protect users' sensitive information such as identity information, location, mobility traces, habits from any other parties [EE12; WW11]. Hence, the privacy challenge is particularly significant in the application layer.

In order to meet these security and privacy requirements, authentication and authorization mechanisms designed for IoT are indispensable. **Authentication** stands for validating whether a given identity of an IoT entity is genuine or not. At the device layer, the authentication of devices is necessary to prevent illegal access, tampering and camouflage. The authentication at the network layer refers to the signaling data integrity protection to avoid DoS attacks. Roughly speaking, the service layer authentication is coupled with the key management system and access control policies whereas the application layer authentication is related to the identification, authentication, and authorization of user identities. **Authorization** refers to granting the corresponding access permissions to the authenticated user identity. In other words, after authenticating the genuine identity of a user, the system grants him/her corresponding rights. Therefore, established identities and authentication protocols are prerequisites for authorization protocols, which logically applied at service and application layers, and are enforced at the device layer.

To sum up, authentication and authorization mechanisms are keystones and used to support IoT security concepts like confidentiality, integrity, availability, and privacy in IoT systems as depicted in Figure 1.2. The authentication is responsible for verifying legitimate users (identities) while authorization process prescribes rules how these users interact with each other. Therefore, in order to enable user-centric security to meet these security requirements in peer-to-peer networks, a resilient and trustworthy identity management framework and a supporting access control framework with dynamic defense are required. The former should be able to identify all IoT entities and authenticate legitimate users without relying on third parties in distributed peer-to-peer networks while the latter should be able to grant corresponding permissions to legitimate users in trustless peer-to-peer networks and dynamic prevent malicious users from breaching the system and stealing sensitive information when the security system is running.

1.4 Research Strategies

In the previous discussion, we sort out the research challenges from the perspective of IoT unconventional characteristics, digital identity (authentication) and

access controls (authorization). Then, we elaborate main security requirements when designing IoT secure infrastructures and illustrate relations between security requirements and authentication/authorization mechanisms.

1.4.1 Research Statement

From summarized IoT challenges and security requirements, we can see, in order to build our IoT secure infrastructure to meet security requirements and take into account mentioned IoT unconventional characteristics, new user-centric identity management and access control frameworks in peer-to-peer networks, need to be proposed. Therefore, we summarize our research problem:

- how to design the identity management framework, which can identify all IoT entities and authenticate legitimate users without relying on third parties.
- how to design the access control framework in trustless peer-to-peer networks, which can grant corresponding permissions to legitimate users and prevent malicious users from breaching the system and stealing sensitive information.

1.4.2 The Methodology of Building the IoT Secure Infrastructure

Table 1.2 lists all the design principles and approaches that we have adopted in building our IoT security infrastructure according to requirements from unconventional characteristics, security, and privacy.

Firstly, we propose to use the blockchain technology to build a blockchain based IdP to eliminate unnecessary third-party identity providers. Compared to traditional identity management systems, the blockchain based IdP is resilient, distributed, and trustworthy:

- every blockchain node follows the formed identity consensus and could be thought of as an IdP, which avoids phishing and single point of failure from the traditional identity solutions (*IDC-B*);
- our proposed blockchain based IdP is scalable (see, scalability analysis in Appendix A) and resides in every possible edge node which makes our IoT secure infrastructure more suitable for dynamic changes IoT environments;

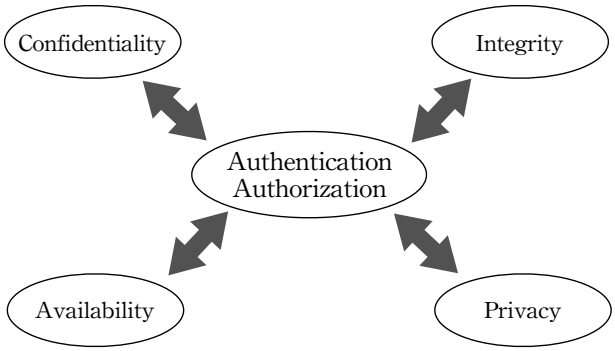


Figure 1.2 – Relationship between security requirements and AA

- users can manage their identities on the blockchain based IdP and do not have to grant full trust to third parties who compromise privacy (*IDC-C*);
- our blockchain based IdP authentication mechanism also supports constrained IoT devices due to the lightweight simplified payment verification bitcoin client which allows users to verify transactions without synchronizing entire block.

Then, we take advantage of the social relationships and a hierarchical identity tree structure to design an IoT social model, as shown in Figure 3.3 and Figure 3.4, which can unify all IoT entities from people, companies, organizations to devices, applications and services (*IDC-A*). In the social model for IoT,

- each individual or collective has an identity as the root of the hierarchical identity tree;
- individuals or collectives need to register their root identities in the blockchain based IdP;
- these root identities can be used to establish relationships between individuals and collectives;
- all things (e.g., devices, services) in IoT are correlated to their owner (namely, individual or collective) through the hierarchical identity structure.

The adoption of social concept in IoT and the global blockchain-based IdP make the IoT extensible and ubiquitous networks in which all heterogeneous social entities (namely, individuals, collectives, things) can freely join and leave based on created hierarchical identities. More importantly, relationships from socialized IoT can be used to design the access control framework (*ACC-A*) in trustless peer-to-peer networks. In detail, relationships between subjects and IoT things allow owners to set security policies to secure their own devices or services, which could integrate all IoT things to subjects with built-in security policies through ownership. Furthermore, in order to enhance the availability of our IoT secure infrastructure, we also introduce an autonomic threat detection mechanism (*ACC-B*) as the complementary of our access control framework, which can continuously monitor anomaly behavior of device or service operations and feed the result to our access control framework.

In addition to giving heterogeneous IoT entities universal identities, we also follow the Service Oriented Architecture (SOA), which could solve the interoperability problem via providing universal interfaces like the REpresentational State Transfer (REST) APIs in the heterogeneous IoT environments. By such, we shape our IoT secure infrastructure into reusable software components that provide reusable security services, including identification services, relationship management services, authentication services, access control services, and threat detection services. These services are defined and developed through a set of universal REST APIs, which are loosely coupled, reusable and composable components for any scenario in the heterogeneous IoT environments. Moreover, in order to improve adaptability to dynamic changes in IoT environments, we take advantage of the asynchronous and non-blocking processing architectures (namely, event-driven model) in our implementation whereby the application thread could continue to execute other processing (non-blocking) even if encountering the time-consuming operations like network input/output, disk input/output, mutual exclusion. The event-driven model makes our IoT secure infrastructure deployed on

Table 1.2 – Methodology of building the IoT secure infrastructure

Requirements	Design Principles	Approaches	Proposed Solutions	Chapters
Scalability	Distributed Architecture; Peer-to-Peer	Blockchain	Blockchain based Identity Provider	4
		Social Concept	Extensible Socialized IoT Networks	4, 5
Heterogeneity	Universal; Modelling	Service-oriented Architecture	RESTful API	6
		Social; Hierarchical Identity Tree	Universally Social Model for IoT	4, 5
Dynamic Changes	Mobility; Real-time Response; Dynamic Adaptability	Blockchain	Global Blockchain Identity Repository	4
		Asynchronous and Non-blocking	Proposed Solution at Edge using Vert.x	6
Limited resources	Considering the Consumption from Computation, Storage, Bandwidths, and Power	Lightweight Client for Authentication	SPV Client for Blockchain Authentication Mechanism	4
Security & Privacy	Confidentiality	Identity and Access Control Management based Security Framework	Data Encryption in Transit & at Rest	4
	Integrity		Data and Device Integrity Protection	
	Availability		Threat Detection	5
	Privacy		Blockchain based Identity Provider	4
			Hierarchical Identity Tree Structure	

edge nodes capable of handling high concurrency, real-time, and dynamic request in IoT environments.

When building our secure IoT infrastructure, the proposed identity and access control management frameworks also lay a solid foundation to achieve security requirements like confidentiality, integrity, availability, and privacy. For example, in chapter 4, we introduce schemes not only to protect data using encryption and integrity check in transit (namely, data transmission in network layer) or at rest (namely, data stored in servers) but to ensure the device integrity through binding identity information and unmodified device attributes. As for privacy, the distributed blockchain based IdP is no longer controlled by third parties, which extremely dissipates privacy consideration of private information being utilized by unauthorized parties. Moreover, the detachment of the root identity for individuals or collectives and partial identity for things is beneficial to achieve anonymity. Users can generate partial identities for each application scenarios and use self-generating partial identities to access services as long as the service provider does not need the endorsement from their root identities.

1.4.3 Overview of the Proposed IoT Secure Infrastructure

We adopt several new ingredients such as the blockchain technology, social IoT, SOA, and asynchronous and non-blocking model to build the IoT secure infrastructure to handle these IoT unconventional characteristics and meet security requirements. Generally speaking, our proposed IoT secure infrastructure consists of two pillars: blockchain-based identity management framework for IoT (BIMSIT) and decentralized access control framework with threat detection (DITAC).

BIMSIT is responsible for managing identities of all IoT entities and provides the fundamental authentication mechanism to authenticate identities without relying on any third parties. In detail, we take advantage of the hierarchical identity tree structure to organize all things falling into each individual or collective. Then, we build a distributed blockchain based IdP by which all individuals or collectives can manage (namely, register, update, revoke) their identities and enable authen-

tication of IoT entities (namely, individuals, collectives, things) without relying on any third parties. More importantly, we formalize our BIMSIT (blockchain identity management and authentication), verify the correctness of the BIMSIT using Burrows–Abadi–Needham (BAN) logic [BAN89], and conduct the security and privacy analysis based on our system and threat model, which proves that our solution is secured from theoretical perspectives.

DITAC reveals how the identity information and relationships are used to design access controls in trustless peer-to-peer networks and how to detect threats at runtime. Specifically, relying on the blockchain based identity framework, DITAC is capable of managing all IoT things falling into the same individuals or collectives through creating identities for these things. Between individuals or collectives, we make use of capability-like relationships to build trust in trustless IoT environments and then grants permissions via mapping users of two participants in the established relationships. In order to detect threats of our IoT secure infrastructure at runtime, we also present autonomic threat detection mechanism which could detect anomalies triggered by any unusual event (e.g., cyber-attacks), and provide the required intelligence so that the access control framework will deal only with their security concerns. The basic idea is to create a reference model for each smart object that describes its normal behavior.

In order to examine these solutions, we present the implementation architecture of our proposed IoT secure infrastructure, which mainly focuses on identity and access management module, blockchain based IdP module, and dashboard module. Based on these module implementations, we use the dashboard GUI to reproduce the case studies proposed in chapter 3 and qualitatively evaluate the performance of BIMSIT and DITAC from the perspective of scalability, heterogeneous, mobility, and limited resources. In the meantime, we also quantitatively analyze our IoT secure infrastructure using some proposed evaluation metrics such as storage, computation, and communication cost. Since threat detection mechanism is a collaboration work with Jesus Pacheco and Victor Hugo Benitez Baltazar, we only provide the experimental and simulation results in the implementation chapter 6. The preliminary experimental results for intrusion detections show that our framework can effectively detect both known and unknown threats with high detection rates and low false positive alarms.

1.5 Dissertation Outline

The dissertation is organized as follows. In chapter 2, we conduct a survey over 41 selected papers and projects according to our proposed criteria and evaluated them from the perspective of the proposed unconventional characteristics and security requirements. Then, we investigate the recent status of the identity and access control management, point out the principles and approaches (blockchain, social IoT, SOA, asynchronous and non-blocking) of building our IoT secure infrastructure and survey the state of the art of applying these approaches to IoT environments. Chapter 3 gives an overview of our proposed IoT security infrastructure, which comprises the identity management framework and access control framework with threat detection functionality. In order to better understand our

proposed solution, we also describe a set of futuristic cases and scenarios in chapter 3. Chapter 4 describes the specific scheme to build a blockchain-based identity management framework which is the critical component to our IoT secure infrastructure. It not only covers the hierarchical identity information management and blockchain based IdP but verify the correctness of the BIMSIT (blockchain identity management and authentication) and conduct the security/privacy analysis based on our system and threat model. On the basis of our blockchain based identity management framework, we propose the decentralized IoT access control framework with the threat detection functionality in chapter 5. We, firstly, supplement many definitions such as *Policies, Domains, Roles, Relationships* in the access control model and then elaborate the specific access control mechanism which shows how relationships are used to realize the access control. At the end of chapter 5, we also introduce the threat detection mechanism to protect the operations of devices against threats. Chapter 6 elaborates the prototyping details and gives the experimental evaluation results. Specifically, we introduce the implementation of our proposed IoT secure infrastructure, which mainly focuses on identity and access management module, blockchain based IdP module, dashboard module, and threat detection module. Making use of these module implementations, we evaluate the performance, reproduce the case studies proposed in chapter 3 and present them using dashboard GUI. Besides, we also test the threat detection module in a smart home testbed. Finally, we summarize and give a future vision of the thesis in chapter 7. Figure 1.3 illustrates the diagram of our dissertation.

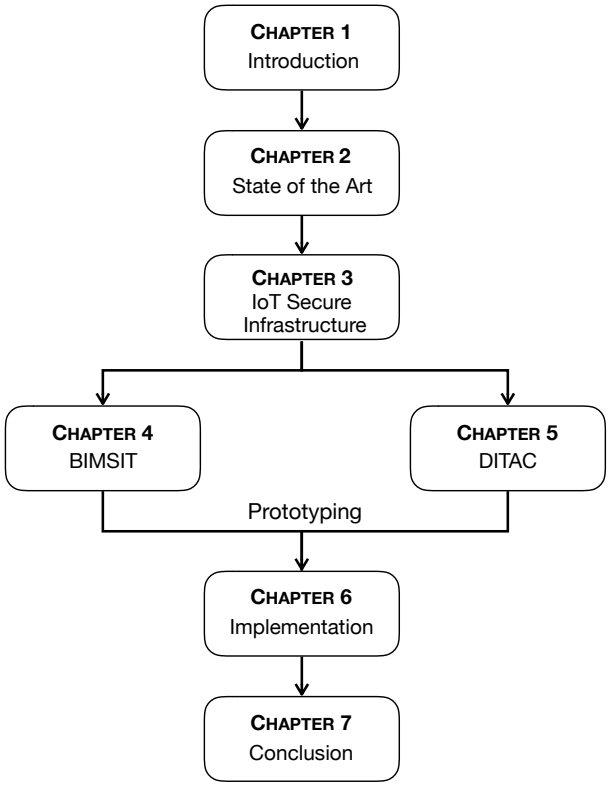


Figure 1.3 – Dissertation outline diagram

1.5.1 List of Publications

- Xiaoyang Zhu and Youakim Badr. "Identity Management Systems for the Internet of Things: A Survey Towards Blockchain Solutions." Sensors, 18(12), p.4215. IF: 2.475.
- Xiaoyang Zhu and Youakim Badr. "A Survey on Blockchain-Based Identity Management Systems for the Internet of Things." Proceedings of the 2018 IEEE Symposium on Blockchain. IEEE, 2018, pp. 1568–1573.
- Xiaoyang Zhu and Youakim Badr. "Fog Computing Security Architecture for the Internet of Things Using Blockchain-Based Social Networks." Proceedings of the 2018 IEEE Symposium on Blockchain. IEEE, 2018, pp. 1361–1366.
- Xiaoyang Zhu, Youakim Badr, Jesus Pacheco, and Salim Hariri. "Autonomic Identity Framework for the Internet of Things." Proceedings of the 2017 IEEE International Conference On Cloud and Autonomic Computing (ICCAC). IEEE, 2017, pp. 69–79.
- Jesus Pacheco, Xiaoyang Zhu, Youakim Badr, and Salim Hariri. "Enabling Risk Management for Smart Infrastructures with an Anomaly Behavior Analysis Intrusion Detection System." Foundations and Applications of Self* Systems (FAS* W), 2017 IEEE 2nd International Workshops On. IEEE, 2017, pp. 324–328.

CHAPTER 2

State of the Art

Contents

2.1 Introduction

As we discussed in the introduction chapter, designing secure components for IoT is faced with many challenges from intrinsic IoT characteristics and users as owners intend to set security policies to control their devices or services instead of relying on some centralized third parties. Therefore, in order to build a user-centric IoT secure infrastructure in peer-to-peer networks, we conduct a survey on some IoT-related review papers [Ban+11; CM12; DPB13; LM14; Fuq+15; Fer15; Vas+15; Sic+15; Raz+16; SS17; Ngu+17; FS17; dCru+18] over the past eight years. Based on these survey papers, we screen forty-one from over 70 IoT solutions according to the criteria described as follows:

- Criteria 1: the surveyed solutions should be designed for IoT environments (e.g., solutions for wireless sensors networks, RFID systems or other IoT devices);
- Criteria 2: these solutions should take care, at least, one of our proposed unconventional characteristics (scalability, heterogeneity, dynamic changes, and limited resources);
- Criteria 3: the security feature is indispensable in our surveyed IoT solutions.

Then, we classified these selected IoT solutions into three categories in terms of academic research papers, open source projects, and third-party cloud service providers. Since the IoT aims at connecting everything ranging from people, organizations, cooperations to devices, applications, services data, these IoT solutions should be good enough for adapting the previous IoT unconventional characteristics namely, scalability, heterogeneity, dynamic changes and limited resources. Moreover, these IoT systems should also take security and privacy as their priority. Thus, we analyze their frameworks from the perspective of our summarized IoT unconventional characteristics (scalability, heterogeneity, dynamic changes, and limited resources), authentication and authorization mechanisms, and security requirements (confidentiality, integrity, availability, and privacy). According to our analysis on these selected IoT systems, we can observe that some principles or approaches such as the degree of decentralization, SOA, and event-driven model, are beneficial to handling these unconventional characteristics in IoT. Besides, building an IoT secure infrastructure to meet security requirements (namely, confidentiality, integrity, availability, and privacy), requires new user-centric identity management and access control frameworks in peer-to-peer IoT networks. The identity management framework is responsible for the identification of all IoT entities and verifying legitimate users in peer-to-peer networks while the access control framework prescribes rules how these users interact with each other and ensures the availability of the secure IoT infrastructure at runtime through threat detection functionalities. Therefore, we also investigate the current status and survey identity and access control management frameworks. The survey turns out that the challenges mainly focus on:

- How to build a trustless identity management framework which can eliminate the unnecessary third-party identity providers to protect the security and privacy of users.
- How to design a new access control framework for trustless environments needs to be proposed to manage access permissions and ensure the availability of IoT systems through actively defending various attacks.

In the following sections, we firstly establish specific metrics to evaluate selected IoT systems. Then, we investigate the current status and survey identity and access control management frameworks. At last, we point out the principles and approaches of building our IoT secure infrastructure and the state of the art of applying these approaches to the IoT environments.

2.2 The Existing IoT Systems

In order to evaluate our selected IoT solutions, we need to quantify the previous proposed unconventional characteristics (namely, scalability, heterogeneity, dynamic changes, limited resources) and security requirements (namely, confidentiality, integrity, availability) in that some of them are too ambiguous to measure. Besides, we also establish metrics to authentication and authorization mechanisms from the perspective of network and application layers respectively.

Scalability. From the scalability proof over identity management frameworks in Appendix A, we can conclude: the higher the degree of decentralization, the better the scalability becomes for the secure IoT infrastructure. Thus, we evaluate scalability according to the degree of decentralization from Centralized (Ce), Decentralized (De) to Distributed (Di).

- Centralized IoT systems refer to systems in which identity providers and service providers are integrated; If users want to access services provided by centralized IoT systems, they need to register their accounts first;
- Decentralized IoT systems stand for systems from which identity providers are detached from service providers. However, users still have to rely on centralized identity services provided by third trusted parties. In other words, if the identity server goes down, no one could prove whether their identities are genuine;
- Distributed IoT systems refer to systems in which users do not need to rely on third-party identity providers and can generate identities by themselves. For instance, recent IoT systems take advantage of the self-sovereign identity (SSI) powered by blockchain technology.

Heterogeneity. The heterogeneity refers to the existence of various network communication protocols and application platforms in cyber and physical systems. Diversity and heterogeneity lead to interoperability problems in IoT environments. On the one hand, all kinds of IoT systems or devices with different communication protocols or standards should be able to communicate with each other. On the other hand, all entities in IoT should identify and verify each other. Therefore, we evaluate the heterogeneity from the following two aspects:

- Network Layer Supported (NLS) heterogeneity means IoT systems could connect devices with different communication protocols or standards such as Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), eXtensible Messaging and Presence Protocol (XMPP) through WiFi, ZigBee, Bluetooth or LoRa;
- Application Layer Supported (ALS) heterogeneity means having high-level interoperability design, and all IoT entities could know each other. The indicators include the Semantic Web, Web of Things and Semantic Ontology.

Dynamic Changes. The dynamic environment needs real-time processing capability in the secure IoT solution. Therefore, the asynchronous and non-blocking processing architectures are essential to building secure IoT systems. Since the event-driven model (E) plays a critical role in asynchronous and non-blocking programming, we use it to evaluate the adaptability to dynamic changes. Besides, the IoT is ubiquitous which means IoT devices, such as vehicles or wear-

able devices are subject to strong mobile capability. Thus, we independently list mobility (M) as another essential feature to evaluate dynamic changes.

Limited Resources. The evaluation of computation, storage, and power consumption is arduous to quantify since these IoT systems are not identical. Even if two implementations of the same IoT framework, they might have different resource consumption due to the two different implementation programming languages. Therefore, we only compare the communication protocols to evaluate whether the framework is more appropriate for resource-constrained devices. If a framework adopts communication protocols like MQTT, CoAP, which are designed for IoT constrained devices, we assume, it highly supports (HS) resources constrained devices. Otherwise, it only has medium support (MS) for resource-constrained devices.

Authentication. The authentication is responsible for identifying legitimate users behind their identities. In other words, identities should include authentication mechanisms so that users can claim their own identities. Therefore, authentication and identity are indivisible. We evaluate the authentication mechanisms of these selected IoT systems from the two aspects:

- Network Layer Supported (NLS) authentication means that the Transport Layer Security (TLS) protocol is supported in the network communication layer;
- Application Layer Supported (ALS) authentication refers to providing identity management framework and authentication mechanisms.

Authorization. After the authentication, the next step is to check if the corresponding user has permission to access requested services. Like the authentication, we propose to evaluate the authorization from the following two aspects:

- Network Layer Supported (NLS) authorization means if the TLS authorization extensions [BH10] is supported in the network communication layer;
- Application Layer Supported (ALS) authorization refers to having a complete application layer authorization mechanism by which users could configure their access control policies.

Confidentiality. By relying on the previous authentication and authorization, confidentiality refers to preventing sensitive data from being retrieved and cracked by other unauthorized parties. Since data could be in transit (network layer communication) or at rest (storage), we could evaluate the confidentiality from the following two aspects:

- Network Layer Supported (NLS) confidentiality means if the TLS protocol is supported in the network communication layer;
- Storage Confidentiality Supported (SCS) refers to having encryption schemes for data at rest to protect the data in storage devices from disclosure and tampering.

Integrity. Protecting data or programs from being tampered by unauthorized users, integrity is often tied to confidentiality protection. Therefore, we could evaluate the integrity from the following aspects:

- Network Integrity Supported (NIS) means if the TLS protocol is supported in the network communication layer;
- Data Integrity Supported (SIS) refers to having integrity protection schemes for data at rest to protect the data in storage devices from tampering.

- Program Integrity Supported (PIS) stands for bootstrapping integrity checking mechanisms of programs or operating systems.

Availability. In addition to the security requirements at design-time (e.g., authentication, authorization, confidentiality and privacy), the IoT systems need to take care of the availability of services at runtime. In other words, IoT systems should be resilient enough to resist various attacks and have countermeasures to mitigating risk and threat. Therefore, we evaluate the availability according to whether IoT systems support threat detection (TDS).

Privacy. There are many techniques for privacy-preserving such as encryption mechanisms (in network layer), anonymity, pseudonymity, multi-party computation and secret sharing in the application layer [GPR12; Gli11; Fuh13]. Since we have already classified the network layer encryption into the confidentiality category, we evaluate the privacy of these IoT systems only from the application layer.

On the basis of these quantified metrics of unconventional IoT characteristics, security requirements, authentication and authorization mechanisms, we evaluate all selected IoT systems according to classified categories: academic research papers, open source projects, and third-party cloud service provider systems in the following sections.

2.2.1 Proposed IoT Systems from Research Papers

According to our selection criteria, we end up evaluating twelve academic papers with respect to IoT unconventional characteristics, security requirements, authentication and authorization mechanisms. The specific analysis is listed in Table 2.1, Table 2.2 and described as follows:

Conzon et al. [Con+12] propose a secure IoT communication framework called VIRTUS, which adopts the event-driven model and Open Services Gateway initiative (OSGi) to manage the dynamic functionality modules at runtime. Also, the flexible and dynamic modules are also contributing to customizing the IoT solutions even for devices with constrained resources (e.g., smartphones, sensors or RFID reader). Considering the security and privacy challenges, VIRTUS takes advantage of the XMPP to ensure the authentication, encryption, and integrity in network communications. Moreover, coupled with some XMPP extensions, the VIRTUS also ensures the scalability and interoperability features. However, VIRTUS does not handle much about authorization and availability. On the one hand, the authorization functionality is optional and only designed for data or access resources rather than for all entities in IoT environments. On the other hand, the service availability in VIRTUS at runtime has not been evaluated through establishing specific threat model and penetration testing.

Parlanti et al. [Par+10] present the Service Application Integration (SAI) system, which relies on the event-driven model (Message Bus) to organize access to heterogeneous data and components in distributed environments. The design principles of event bus and service-oriented model attempt to handle the scalability, interoperability and dynamic changes in IoT environments. Although having the back-end security manager for authentication, authorization, and network confidentiality (integrity), the SAI system does not provide more deeply security

Table 2.1 – Summary of academic IoT systems: unconventional characteristics

	Scalability	Heterogeneity	Dynamic Changes	Limited resources
VIRTUS	De	NLS	E	HS
SAI	De	ALS	E	NI
SMEPP	De	NLS, ALS	E, M	HS
Sensor Andrew	De	NLS, ALS	E	HS
Otsopack	NI	ALS	M	MS
STFTP	NI	NI	NS	NI
Ferreira	De	ALS	NS	MS
NOS	De	NLS, ALS	E, M	HS
SOCRADES	NI	ALS	E	MS
IoT-MP	De	NLS, ALS	E, M	HS
SensorAct	De	ALS	E, M	MS
DREMS	NI	ALS	M	NI
Legend No Information (NI) Not Supported (NS) Supported (S)	Centralized (Ce) Decentralized (De): Client-Server Model Distributed (Di): P2P SSI + decentralized AC	Network Layer Supported: multi communication protocols (NLS) Application Layer Supported (ALS): Semantic, REST	Mobility (M) Event -driven (E)	REST, Web based: medium support (MS) CoAP, MQTT: highly support (HS)

analysis on these established security requirements and not involve availability and privacy.

Caro et al. [Car+09; RLN11] introduce a Secure Middleware for Embedded Peer to Peer (P2P) (SMEPP) system, which uses the service-oriented architecture (SOA) to design an adaptive, scalable and interoperable framework for devices in the SMEPP system. Besides, security functionalities including authentication, and network confidentiality (integrity), as critical components, are dealt with in the SMEPP system. However, some other security requirements like authorization, availability, and privacy are not covered.

Rowe et al. [Row+11] build Sensor Andrew for sensors and actuators in large-scale and heterogeneous environments. It takes advantage of XMPP and XMPP Extension Protocols and introduces many features such as event driven-model, authentication, authorization, and network confidentiality (integrity). Nevertheless, the availability of the Sensor Andrew is not appropriately evaluated. Besides, privacy and authorization are also restricted in the network communication layer.

Gómez-Goiri et al. [Góm+14] present a modular and straightforward semantic framework, namely Otsopack, to handle interoperability issues in heterogeneous IoT environments via defining a set of REST interfaces. Otsopack adopts the Triple Space Computing (TSC) paradigm, which is a shared memory method of facilitating indirect communication between heterogeneous applications [HG10]. Enabling security is another essential feature in the Otsopack, which is built on OpenID-based authentication and authorization solutions. However, there are still some other security requirements, such as availability, privacy, confidentiality, and integrity that are not included.

Isa et al. [Isa+12] propose an enhanced security file transfer protocol called STFTP for embedded devices (e.g., WiFi Access Points, Remote Base Stations), in which the authors integrate the Trivial File Transfer Protocol (TFTP), Diffie-Hellman Key Exchange (DHKE) and Advanced Encryption Standard (AES) to provide authentication, authorization, and confidentiality at the network layer. By using the STFTP, the system kernel upgrading and patching process of embedded devices could be protected from eavesdropping and tampering of malicious attackers. Similarly, Ferreira et al. [Fer+14] present a secure deployable

Table 2.2 – Summary of academic IoT systems: security requirements

	Authentication	Authorization	Confidentiality	Integrity	Availability	Privacy
VIRTUS	NLS	NS	NLS	NIS	NS	NS
SAI	NLS, ALS	NLS, ACP	NLS	NIS	NS	NS
SMEPP	NLS, ALS	NS	NLS	NIS	NS	NS
Sensor Andrew	NLS	NLS	NLS	NIS	NS	NS
Otsopack	NLS, ALS	NLS, ACP	NS	NS	NS	NS
STFTP	NLS	NLS	NLS	NIS	NS	NS
Ferreira	NLS	NLS	NLS	NIS	NS	NS
NOS	NLS, ALS	NLS	NLS, SCS	NIS SIS	TDS	NS
SOCRADES	NLS, ALS	NLS, ACP	NLS	NIS	NS	NS
IoT-MP	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
SensorAct	NLS, ALS	NLS, ACP	NI	NI	NS	S
DREMS	NLS, ALS	NLS, ACP	NLS	NIS	NS	NS
Legend No Information (NI) Not Supported (NS)	Network Layer TLS/HTTPS Supported (NLS); Application Layer Supported (ALS)	TLS/HTTPS (NLS) Access Control Policies (ACP)	Network Layer Supported TLS (NLS); Storage Confidentiality (SCS)	Network Integrity Supported (NIS); Data Integrity (SIS); Program Integrity (PIS)	Threat Detection Supported (TDS)	Supported (S)

and transparent framework combining the AES, TLS, and OAuth 2.0 Protocol to ensure the authentication, authorization, and network confidentiality (integrity). Besides, they provide the Universal Plug and Play (UPnP) and REST APIs to interconnect heterogeneous IoT devices at the application layer. However, STFTP and Ferreira do not consider the availability of IoT systems through threat detection and privacy-preserving from users' perspective.

Sicari et al. [Sic+16] introduce NetwOrked Smart objects (NOS), a lightweight middleware with designed functionalities (e.g., data caching, in-memory processing) to bridge heterogeneous devices or services in dynamic IoT environments. NOS defines the NOS data format with many security considerations such as authentication, authorization, confidentiality, and integrity. More importantly, it ensures availability over various attacks like packet sniffing, man-in-the-middle attack, and identity spoofing.

De Souza et al. [De +08] present a Web service-oriented IoT infrastructure (namely, SOCRADES) for the shop floor in smart manufacturing. It divides the architecture into three types of services: device services, cross-layer services, and application services. The SOCRADES middleware adopts the asynchronous event-driven model and provides universal SOA interfaces. Thus, it supports interoperability, dynamic changes, and limited resources. However, it lacks support for the scalability issues in that the system of shop floors is a relatively closed system. Besides, the Web service based security in the SOCRADES only considers authentication, authorization, network confidentiality (integrity), and no support for privacy preserving and threats detection.

Elkhodr et al. [ESC16] propose the Internet of Things Management Platform (IoT-MP) to manage massive, heterogeneous and resource-constrained devices with the consideration of privacy preserving. The IoT-MP presents a distributed two-tier architecture: the distributed local IoT management tier and the centralized cloud IoT management tier. The tiered architecture classifies devices into Managed Things (MT), Managers and the Manager of Managers (MoMs), which makes the platform more scalable. Moreover, the security and privacy modules in the IoT-MP provide authentication, authorization, network confidentiality (integrity), and privacy protection functionalities for users.

Arjunan et al. [Arj+15] introduce SensorAct, a decentralized and event-based IoT framework for smart buildings, in which all devices are abstracted and correlated with the Virtual Personal Device Servers (VPDS). The VPDS are managed by their owners who need to register their accounts in the trusted Brokers responsible for maintaining a registry of users. The security aspects in the SensorAct [Arj+12], only cover authentication, authorization, and privacy.

Levendovszky et al. [Lev+14] propose a model-driven Distributed Real-time Managed System (DREMS), which provides a set of tools for the IoT application development ranging from modeling, debugging, testing to runtime deploying and monitoring. The security framework in the DREMS only focuses on authentication, authorization and network confidentiality (integrity) functionalities. However, the authorization mechanism follows the label checking based multi-level security on limiting the information flows according to different information classification levels.

From Table 2.1 and Table 2.2, we observe that IoT systems from academic research papers do not consider all proposed IoT unconventional characteristics and security requirements but mainly focus on some specific points. For instance, Sicari et al. [Sic+16] have a comprehensive interoperability framework for heterogeneous devices or services in IoT environments; Isa et al. [Isa+12] only focus on the secure file transfer protocol.

2.2.2 Open Source IoT Projects

IoT is regarded as a global infrastructure for the information society, which involves various manufacturers, vendors, communities and standard groups. Consequently, it is a mission impossible to unify all parties only relying on closed IoT systems. In this section, we select and evaluate representative open source projects from the perspective of IoT unconventional characteristics, security requirements, authentication and authorization mechanisms. The specific analysis is listed in Table 2.3, Table 2.4 and described as follows:

Hydra (LinkSmart) project [ERA09] funded by the European Union (EU) creates a scalable platform using concepts such as event-based model, SOA and semantic-based model to unify physical heterogeneous IoT devices in ambient intelligence systems. It makes use of the semantic and ontology module to provide an interoperable environment for heterogeneous devices and applications. However, the security model in Hydra relies on Extensible Markup Language (XML) security standard [Dou02], which causes many issues [FS17] such as low efficiency to constrained devices, and the incomplete of key and certificate management. Besides, Hydra only offering the authentication and network confidentiality (integrity), it does not provide other security functionalities (e.g., authorization, availability and privacy).

The IoT-A architectural reference model [Ser+13] proposed at the EU 7th Framework Programme Research Project (2007-2013), gives a top-level description over the entire IoT domain ranging from the requirement analysis of concrete application scenarios to the design of concrete IoT reference models. It not only considers the connectivity of various heterogeneous devices and service abstraction management at the high level but addresses many of security requirements

Table 2.3 – Summary of open source IoT systems: unconventional characteristics

	Scalability	Heterogeneity	Dynamic Changes	Limited resources
Hydra (Linksmart)	De	NLS, ALS	E	HS
IoT-A	De	NLS, ALS	M	HS
OpenIoT	De	NLS, ALS	M	HS
BeTaas	De	NLS, ALS	E	HS
IoT@Work	De	NLS, ALS	E	HS
FiWare	De	NLS, ALS	E, M	HS
Devicehive	De	NLS	E, M	HS
Kaa	De	NLS, ALS	E, M	HS
Sitewhere	De	NLS, ALS	E, M	HS
Webinos	Di	ALS	E, M	MS
oneM2M	De	NLS, ALS	E, M	HS
&Cube	De	NLS, ALS	E, M	HS
iCore(SecKit)	De	ALS	E, M	MS
Butler	De	NLS, ALS	E, M	HS
Alljoyn	Di	NLS, ALS	M	HS
IoTivity	De	NLS, ALS	M	HS
WSO2	De	NLS, ALS	E, M	HS
Legend No Information (NI) Not Supported (NS) Supported (S)	Centralized (Ce) Decentralized (De): Client-Server Model Distributed (Di): P2P SSI + decentralized AC	Network Layer Supported: multi communication protocols (NLS) Application Layer Supported (ALS): Semantic, REST	Mobility (M) Event -driven (E)	REST, Web based: medium support (MS) CoAP, MQTT: highly support (HS)

and gives the corresponding countermeasures. It includes the AuthN module for authentication, AuthZ module for authorization, Key Exchange and Management (KEM) module for network security (confidentiality and integrity), and even Pseudonymization (PN) module for privacy. In other words, security requirements at design time have been considered except the system availability evaluation with regard to malicious attacks at runtime. Based on the architectural reference model of IoT-A, the OpenIoT project [KL14] instantiates the reference model to provide an open source implementation, which has capabilities of integrating submodules from other projects like the Global Sensor Networks (GSN) [AHS06] to compensate the lack of security considerations. Another related project is the Building the environment for the Things as a Service (BeTaaS) platform [Val+16], which improves the IoT-A references model and virtualizes IoT devices through 3 layers: physical adaptation layer, logical layer, and service layer.

IoT@Work [GPR12; HP +13], takes advantage of the event-driven model to build the Event Notification Service (ENS) for their flexible and pluggable automation middleware in the manufacturing domain. As for security aspects, it adopts the Extensible Authentication Protocol (EAP) to ensure the authentication, confidentiality, and integrity of the communication in the network layer [FG12]. Furthermore, a capability-based access control mechanism [GPR13] is developed to support the fine-grained authorization, in which anonymous capabilities are introduced to protect the privacy of users. However, they did not cover the availability analysis at runtime.

Fiware [Gli11] builds an open and public infrastructure with the high Quality of Service (QoS) and security guarantees for the Future Internet. It consists of seven major functional building blocks called Generic Enablers (GEs): IoT Service Enablement, Data/Context Management, Security, Web-based User Interfaces, Middleware and Interfaces for Network and Devices, Services and Data Delivery, and the Cloud Hosting. Fiware also adopts the event-driven model,

Table 2.4 – Summary of open source IoT systems: security requirements

	Authentication	Authorization	Confidentiality	Integrity	Availability	Privacy
Hydra (Linksmart)	NLS, ALS	NS	NLS	NIS	NS	NS
IoT-A	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
OpenIoT	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
BeTaas	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
IoT@Work	NLS	NLS, ACP	NLS	NIS	NS	S
FiWare	NLS, ALS	NLS, ACP	NLS	NIS	TDS	S
Devicehive	NLS	NLS	NLS	NIS	NS	NS
Kaa	NLS	NLS	NLS	NIS	NS	NS
Sitewhere	NLS, ALS	NLS, ACP	NS	NS	NS	NS
Webinos	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	TDS	S
oneM2M	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
&Cube	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
iCore(SecKit)	NLS, ALS	NLS, ACP	NLS	NIS	TDS	S
Butler	NLS, ALS	NLS, ACP	NLS	NIS	NI	NI
Alljoyn	NLS, ALS	NLS, ACP	NLS	NIS	NS	S
IoTivity	NLS, ALS	NLS, ACP	NLS	NIS	NS	NS
WSO2	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	NS	NS
Legend No Information (NI) Not Supported (NS)	Network Layer TLS/HTTPS Supported (NLS); Application Layer Supported (ALS)	TLS/HTTPS (NLS) Access Control Policies (ACP)	Network Layer Supported TLS (NLS); Storage Confidentiality (SCS)	Network Integrity Supported (NIS); Data Integrity (SIS); Program Integrity (PIS)	Threat Detection Supported (TDS)	Supported (S)

which is an vital feature not only for interoperability and dynamic changes but for ensuring security functionalities in distributed environments. The generic security enabler includes several basic security services: Security Monitoring, which covers events tracking, risk analyzing, attack monitoring; Context based Security and Compliance, which allows users to configure their security policies at runtime; Identity and Access Control, which is responsible for authentication, authorization, privacy and network confidentiality (integrity). Furthermore, the generic security enabler is designed to be extensible for integrating supplementary security mechanisms.

Devicehive [Dat13], created by DataArt, aims at building a scalable and interoperable IoT data platform using microservices and event-driven model. It provides not only three types of interfaces to connect devices: REST, WebSockets and MQTT but many integration options with other IoT platforms like Amazon Alexa and data analytic engines such as Spark, Cassandra, and ElasticSearch. However, authentication, authorization, confidentiality, and integrity are merely ensured via the JSON Web Token (JWT) in the network layer in the Devicehive without considering attack evaluation and privacy protection.

Kaa [Kaa14] Banana Beach, maintained by KaaIoT, provides the unlimited number of connected devices, real-time device monitoring, rich data analysis functionalities and uses the microservice architecture to build a modular and interoperable IoT platform of integrating all heterogeneous things in IoT environments. Nevertheless, the security and privacy design in Kaa are not sufficiently taken into account, and only the authentication, authorization, confidentiality, and integrity in the network layer are supported through the TLS protocol.

SiteWhere [Sit15] provides an industrial grade and microservice-based IoT application platform, in which users could manage not only devices with various communication protocols like the MQTT, Advanced Message Queuing Protocol (AMQP) and Streaming Text Oriented Messaging Protocol (STOMP) through REST interfaces but asset attached with people, geospatial sites and things in the physical world. More importantly, SiteWhere provides identity management

for human. However, SiteWhere only focuses on authentication and authorization through the administrator adding users with corresponding permissions.

Webinos [Fuh13; FLF12; Lyl+12], tries to create a cross-device IoT application platform, in which the user-centric Personal Zone (PZ) concept is proposed to logically group devices into different authority domains and hence to manage security policies in distributed environments. Furthermore, certificate exchange protocols are presented to solve the identity issue of no trust between different users (Personal Zone Hub) in distributed IoT environments. More importantly, the security framework of the Webinos in [Lyl11] covers all security requirements including authentication, authorization, confidentiality, integrity, availability, and privacy. Albeit, they consider all security requirements and use the self-signed certificates to securely manage devices, their identity management of users relying on OpenID [RR06] and OAuth [Har12] protocols, cannot solve the non-repudiation problem. In other words, it is tough to establish trust between different users, especially for strangers in trustless IoT environments.

OneM2M [Swe+14] as an international standardization organization established in 2012, creates a scalable and interoperable framework for Machine-to-Machine (M2M) communications. The OneM2M organization has 5 working groups focusing on requirements, architecture, protocols, security, abstraction, and semantics respectively, which is similar to the organizational structure of IoT-A [Ser+13] with highly similar reference architecture. The software architecture specification document of oneM2M specifies the reference model from three functional layers: Underlying Network Services Layer, Common Services Layer and Application Layer. Supporting two types of message sequence: synchronous and asynchronous modes that could be deployed according to different use cases, the oneM2M integrates constrained heterogeneous devices through various communication protocols such as MQTT, CoAP and Hypertext Transfer Protocol (HTTP) (Websocket or RESTful APIs). The security framework described in the oneM2M Technical Specification(TS3), provides the generic design principles which involve many aspects from identity management, authentication, authorization to sensitive data privacy protection. Under the umbrella of the oneM2M standards, researchers from academia have extended oneM2M with the high-level semantic framework for interoperability in [LYL14; Ala+15; Dat+15], the mobile crowd sensing IoT framework in smart cities [Dat+16], &Cube project [Yun+15] trying to build the software application platform over consumer electronics.

The iCore project [Gia13; Vla+13], introduces the Cognitive Management Framework (CMF) to solve the heterogeneity problem in IoT environments, in which all real-world objects are abstracted as Virtual Objects (VO) and have capabilities of self-composite to meet application requirements and self-reconfigure to handle dynamic changes of concrete scenarios via the concept of the Composite Virtual Object (CVO). The security of the iCore relies on a model-based security toolkit called SecKit [Nei+14], which not only covers authentication, authorization, network confidentiality (integrity) and privacy but takes into account the risk management through evaluating the vulnerabilities and mapping countermeasures to rules to mitigate threats. Furthermore, others also could propose the supplementary solutions for IoT based on the CMF and SecKit like the DIAT in [Sar+15].

The BUTLER [BUT14], is a location and context awareness IoT framework, which emphasizes on the smart resource and security management in the ubiquitous IoT environments. The BUTLER architecture is divided into three modular layers: communications layer, data/context management layer and services layer, which could be customized and deployed to different devices such as smart object platform installed on gateways (namely, sensiNact [BUT15]), smart mobile platform deployed on smartphones and smart server platform built on servers. The security service in the BUTLER allows users to manage their profiles in distributed applications, which implies the authentication, authorization and network confidentiality (integrity) should be supported.

AllJoyn [All16], managed by the AllSeen Alliance, attempts to develop an interoperable framework in P2P networks across heterogeneous Operating Systems (OS) such as Windows, OS X, iOS, Linux, Android, Unix-like OS. Alljoyn has two types of devices: the AllJoyn Standard Devices with the AllJoyn Standard Application and AllJoyn Routers installed on resource-rich systems (e.g., Android, iOS, Windows) and AllJoyn Embedded Devices only with the AllJoyn Thin Application installed on constrained resource systems (e.g., ThreadX, Arduino). Albeit, having the consideration for limited resources and mobility, the Alljoyn is not good at handling dynamic changes due to its blocking-concurrent model. The security manager from the Alljoyn Security 2.0 [TK16] is a certificate-based service which could ensure the authentication, authorization, network confidentiality (integrity) and privacy preserving. However, security managers in the Alljoyn could be deployed by anyone, which arouses the identity and trust problem. Specifically, the Alljoyn still relies on other identity management solution like OpenID [RR06]. Otherwise, it cannot solve the non-repudiation problem just like in the Webinos where secure communication is confined in each trusted domain. Besides, without considering the human factor, the management job will become burdensome when the number of devices increases rapidly.

IoTivity project [Sub15; Dan+17] sponsored by the Open Connectivity Foundation (OCF), is an open source implementation of the OCF specification, which tries to provide the connectivity solution of integrating all heterogeneous devices. The IoTivity architecture comprises three layers: the connectivity abstraction layer which is responsible for the adaptation of various communication protocols; the resource layer which is in charge of defining the resource model to provide universal interfaces to the upper layer; the application layer in which developers could use the interfaces of the resource layer to develop applications for different kinds of IoT scenarios. Besides, the security mechanisms in IoTivity are composed of the Secure Resource Manager(SRM) and the Session Protection. The former adopts the Access Control List (ACL) to manage access control policies of the abstract resources while the latter uses the TLS and Datagram Transport Layer Security (DTLS) to secure the confidentiality and integrity of network communication.

WSO2 [WSO] proposed an IoT reference architecture [Fre14], which consists of several components such as device, communication, aggregation, event processing, identity and access control management. All devices could be connected with the enterprise event bus or the message broker through REST APIs, MQTT, XMPP or Websockets. The identity and access control management in the WSO2 IoT ref-

erence architecture lays a solid foundation of building other security mechanisms like confidentiality and integrity.

From Table 2.3 and Table 2.4, we see that the selected open source projects are more comprehensive compared with the previous IoT systems from academic research papers. Notably, they pay more attention to the security aspects including authentication, authorization, confidentiality,integrity, privacy and even runtime availability evaluation of the IoT systems such as [Fuh13; Gli11; Gia13]. However, although there are some projects (e.g., Fiware [Gli11]) that have considered many unconventional characteristics and security requirements, confidentiality and integrity are restricted to the network communication layer. These decentralized identity frameworks still rely on third-party centralized identity providers. According to the scalability proof proposed in Appendix A, the IoT systems like Webinos [Fuh13] and Alljoyn [All16] with the distributed identity management are scalable. Nevertheless, these distributed systems cannot solve the non-repudiation identity problem since, in theory, anyone can set up a so-called "trusted" identity provider.

2.2.3 IoT Systems from Third-party Cloud Service Providers

The third category evaluates the popular Cloud-based IoT systems from business companies like Google, Amazon, Microsoft, IBM. The specific analysis is listed in Table 2.5, Table 2.6 and described as follows:

ARTIK [Sam] is an IoT Cloud service developed by SAMSUNG, which supports device, user, data management, application enablement and could integrate with other cloud platforms like Amazon or Google. Integrating various communication protocols such as MQTT, CoAP, REST or Websockets, the ARTIK allows users to manage registered devices through the cloud or the Lightweight Machine to Machine (LWM2M) protocol from portable devices like smartphones. It takes advantage of the Certificate Authority (CA) based certificates produced by manufacturers to manage the device registration and secure communication channels between endpoints. Therefore, the security framework only covers the authentication, network authorization and network confidentiality (integrity) using CA certificates and the DTLS protocol.

Google Cloud IoT [Gooc] is a complete set of solutions for intelligent IoT services from data collection and analysis to decision making. The reference architecture of the Google Cloud IoT focuses on two tiers: the Edge IoT Core [Goob] which is responsible for securely interconnecting all kinds of IoT devices to the Cloud and the Cloud IoT Core [Gooa] which is in charge of data analysis. However, the security mechanism in the Google Cloud IoT is heavily relying on the security services provided on the Cloud like the Google Cloud Identity and Access Management framework while the security of the Edge only provides the basic authentication and communication security between devices through signed CA certificates and the TLS protocol. In March 2018, Google acquired an enterprise IoT platform called Xively [Xiv] which had been integrated into the Google Cloud IoT.

Amazon Web Services (AWS) IoT [Ama] proposes a hierarchical solution from device tier, the communication edge to the cloud computing tiers, which currently

Table 2.5 – Summary of third-party Cloud IoT systems: unconventional characteristics

	Scalability	Heterogeneity	Dynamic Changes	Limited resources
Artik Samsung	Ce	NLS, ALS	E, M	HS
Google IoT Cloud	Ce	NLS, ALS	E, M	HS
Xively	Ce	NLS, ALS	E, M	HS
Amazon IoT	Ce	NLS, ALS	M	HS
Microsoft Azure IoT Suite	Ce	NLS, ALS	E, M	HS
Carriots	Ce	NLS, ALS	E, M	HS
Cloudplugs	Ce	NLS, ALS	E, M	HS
EVRYTHNG	Ce	NLS, ALS	E, M	HS
Losant	Ce	NLS, ALS	E, M	HS
Telit IoT platform	Ce	NLS, ALS	E, M	HS
Ubidots	Ce	NLS, ALS	E, M	HS
IBM Watson IoT	Ce	NLS, ALS	E, M	HS
Legend No Information (NI) Not Supported (NS) Supported (S)	Centralized (Ce) Decentralized (De): Client-Server Model Distributed (Di): P2P SSI + decentralized AC	Network Layer Supported: multi communication protocols (NLS) Application Layer Supported (ALS): Semantic, REST	Mobility (M) Event -driven (E)	REST, Web based: medium support (MS) CoAP, MQTT: highly support (HS)

encompasses eight services or components including Amazon FreeRTOS [AWSa], AWS Greengrass [AWSb], AWS IoT Core [AWSe], AWS Device Management, AWS IoT Device Defender, AWS IoT Analytics, AWS IoT 1-Click [AWSc] and AWS IoT Button [AWSd]. The centralized, hierarchical architecture from the Cloud, edges to devices ensures the prompt response from the edge and provides a powerful data analysis capability in the Cloud. Users could use MQTT, REST and Websokets protocols to connect IoT devices to the edge. Also, the security framework in the AWS IoT is rooted in these three tiers from the secure deployment of devices, communication security to the device abnormal behaviors analysis through data mining in the cloud. However, the privacy of users in such Cloud-based IoT solutions remains a significant challenge in that users have to put all the trust to their service providers.

Microsoft Azure IoT [Mic] describes IoT applications as collecting data through things (or devices), generating insights based on the collected data and taking actions according to the generated insights to improve services. The Azure IoT tries to build a microservice based cloud IoT application framework composed of subsystems which are scalable and easy to deploy independently. The core subsystems mainly involve device management, cloud gateway (IoT Hub), data stream processing and user interfaces. Specifically, the Azure centralized Cloud-based IoT framework takes into account the interoperability problem and, for instance, supports REST, Websockets and MQTT communications between devices and the IoT Hub. The cloud gateway is designed to be event-driven architecture, which could rapidly adapt to the dynamic environment and save energy in resource constrained devices. Furthermore, the security requirements as the critical cross-cutting needs are taken into consideration in design time and runtime including authentication, authorization, confidentiality, integrity, and availability.

Carriots [Alt] maintained by Altair Engineering, provides a Cloud-based IoT application enablement platform, on which developers could build their IoT applications by connecting devices, collecting data, managing devices and data, defining rules in the application to regulate devices, testing and running the applications. Through REST APIs or the MQTT protocol, devices could be connected

Table 2.6 – Summary of third-party Cloud IoT systems: security requirements

	Authentication	Authorization	Confidentiality	Integrity	Availability	Privacy
Artik Samsung	NLS, ALS	NLS	NLS	NIS	NS	NS
Google IoT Cloud	NLS, ALS	NLS, ACP	NLS	NIS	NS	NS
Xively	NLS, ALS	NLS	NLS	NIS	NS	NS
Amazon IoT	NLS, ALS	NLS, ACP	NLS	NIS	TDS	NS
Microsoft Azure IoT Suite	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	TDS	NS
Carriots	NLS, ALS	NLS, ACP	NLS	NIS	NS	NS
Cloudplugs	NLS, ALS	NLS	NLS, SCS	NIS, SIS	NS	NS
EVRYTHNG	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	TDS	NS
Losant	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	NS	NS
Telit IoT platform	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	NS	NS
Ubidots	NLS	NI	NLS	NIS	NS	NS
IBM Watson IoT	NLS, ALS	NLS, ACP	NLS, SCS	NIS, SIS	TDS	NS
Legend No Information (NI) Not Supported (NS)	Network Layer TLS/HTTPS Supported (NLS); Application Layer Supported (ALS)	TLS/HTTPS (NLS) Access Control Policies (ACP)	Network Layer Supported TLS (NLS); Storage Confidentiality (SCS)	Network Integrity Supported (NIS); Data Integrity (SIS); Program Integrity (PIS)	Threat Detection Supported (TDS)	Supported (S)

to the Cloud listeners which are event-based data stream service in dynamic IoT environments. The security in Carriots mainly covers two aspects: communication security (network confidentiality and integrity) using the TLS protocol and the cloud account security using Two Factor Authentication. Users still could create a simple access control list in the Carriots platform to manage their access control policies. However, the availability evaluation of attacks and privacy are not considered in the Carriots.

CloudPlugs [Clob] provides a range of IoT solutions from the edge (Edge One), the communication agent (SmartPlug) to the cloud platform (CloudPlugs IoT Cloud). Based on these components, the CloudPlugs also delivers a set of Industrial IoT (IIoT) products [Cloa] like IIoT Thing, IIoT Connector, IIoT Gateway, and IIoT Supervisory Control and Data Acquisition (SCADA) Gateway. The cloudPlugs adopts the publish and subscribe event-driven architecture, in which things and applications can subscribe to the same communication channel to share messages through REST, Websockets or MQTT. However, due to the lack of references, we could merely find, the security of the CloudPlugs only support the confidentiality, integrity, authentication and network authorization using the TLS protocol.

EVRYTHNG [EVR] is a Cloud-based IoT smart platform, providing the identity management of smart products and real-time data management. Any smart product from tags(QR, RFID) to sensors and chips could be interconnected through the product connection management service which provides the real-time connectivity using event-driven architecture and supports many protocols like MQTT, CoAP, and Websockets. Users also could use the local cloud gateway (namely, THNGHUB) to locally manage smart devices in various IoT scenarios with low latency. Understanding the importance of security, EVRYTHNG provides the enterprise-grade security methodology comprising building blocks: authentication, encryption, access control data retention and infrastructure, which cover the authentication, authorization, confidentiality, integrity, and availability except for the privacy.

Losant [Losb] aims at creating a hardware-agnostic IoT cloud platform for the enterprise environment, by which heterogeneous sensors, actuators, controllers, and machines could be integrated across disparate systems and technical stan-

dards. The Losant mainly builds an enterprise IoT platform from the following 5 aspects: edge computing, data and device management, data visualization, visual workflow engine, and end-user experiences. It not only ensures the communication security through TLS protocol but emphasizes storage confidentiality and integrity using industry-standard encryption mechanism [Losa]. Losant end-user experiences allow users to connect the platform through secure and customized API (authentication), to manage the identity of users and devices with specified access control policies (authorization).

Telit [Tel] creates an enterprise-grade IoT application enablement platform including data capture, edge computing, device management, Cloud connectivity, seamless integration, Web-based and mobile applications. Users could use REST or the MQTT protocol to connect devices with the subscription-based cloud platform. Security features in Telit IoT platform involve the data encryption in transit and at rest, the secure communication using TLS protocol, the session management related to identity and the permission management related to access control, which cover authentication, authorization, confidentiality, and integrity.

Ubidots [Ubi] provides a Cloud-based platform for IoT applications. Following the four steps: device onboarding, data transformation, data management, and data visualization, developers could use the platform to build their Cloud-based IoT applications. Devices could be connected to the event-driven IoT platform through the REST or the MQTT protocol. However, the security in the Ubidots only considers the authentication, confidentiality, and integrity at the network communication layer.

IBM Watson IoT [IBM] proposes a smart and scalable IoT cloud platform with built-in security. Developers could deploy deep learning services not only in the cloud but at the edge. Devices including sensors or gateways could be connected with the platform through the MQTT protocol, and users or developers could use the secured REST APIs to manage devices, applications, and data. The security framework in IBM Watson IoT comprises modules such as the IBM Cloud Identity and Access Management, Threat Intelligence Detection, Security Controls over Users, Applications and Gateways, which cover authentication, authorization, confidentiality, integrity, and availability.

From Table 2.5 and Table 2.6, we observe that IoT systems from third-party Cloud service providers are more mature and comprehensive compared with previous surveyed IoT systems from research papers and open source projects. Some of them not only provide the device connectivity management but bring the enterprise-grade data analysis management in the Cloud or even at the edge devices such as IBM Watson IoT [IBM]. IoT systems from third-party Cloud service providers cover almost every aspect from IoT unconventional characteristics to security requirements. However, the centralized Cloud-based IoT systems still have scalability problems, and the biggest challenge is the privacy problem in that users have to put all the trust to their Cloud service providers.

In summary, IoT systems from academic research papers do not consider all proposed IoT unconventional characteristics and security requirements but mainly focus on some specific points; Most of the open source IoT systems still rely on third-party decentralized identity providers such as OpenID [RR06], which

cannot solve the non-repudiation identity problem since, in theory, anyone can setup a so-called "trusted" identity provider; Third-party Cloud-based IoT systems have the scalability and privacy problems in that users have to put all the trust to their Cloud service providers. Besides, many Cloud-based IoT systems will lead to interoperability problems. Therefore, some principles or approaches such as the distributed and trustless frameworks, SOA, and event-driven model, are beneficial to handling these unconventional characteristics, especially scalability and dynamic changes in IoT when building secure IoT infrastructures. More importantly, trustless IoT secure infrastructures, specifically, trustless identity providers in distributed peer-to-peer networks, can eliminate trust to some unnecessary third-party identity providers, which greatly preserves privacy from users' perspective and brings user-centric security into a reality.

2.3 State of the Art of Digital Identity

The Internet of Things aims at connecting everything ranging from individuals (human-beings), collectives (homes, organizations, companies) to things such as objects from physical and cyber worlds) [ITU12a]. Given billions of people, trillions of IoT devices, and innumerable data resources, the major challenge is how to identify these entities uniquely and how to allocate digital identities to individuals and things through interconnected networks so IoT entities could be easily identified and communicate with each other. Without digital identities, entities could barely transact with others, leading to untrusted environments and consequently the lack of business opportunities.

In the Internet era, the digital identity remains the keystone of online services and upon which security mechanisms (i.e., authentication, authorization, secure exchanges) and protocols are built. As outlined by the International Telecommunication Union (ITU), "an identity refers to a set of information used for uniquely identifying an entity in a given context" [ITU09] whereas an Identity Management (IdM)System refers to the management of identity information through a set of operations, including registering, updating, revoking and looking-up digital identities. However, existing identity management systems in the context of the Internet could not be directly transplanted to IoT environments due to some inherent IoT characteristics like scalability, interoperability, mobility, limited computational and storage resources. Traditional centralized identity management systems, relying on the so-called trusted third parties, raise many privacy concerns (e.g., the Equifax Data Breach [CSP17]). The proliferation of online identity providers leads to fragmented identities scattered all over the Internet, which makes us be overwhelmed by multiple accounts and expose personal information retained by identity providers to vulnerabilities and data breaches. Moreover, fragmented identities from different security domains immensely increase the cost of identity identification and communication.

In this section, we firstly explore the origin of identities from the philosophy of logic and point out the root of many problems in current identity management systems. Then, we survey the art of digital identities in the Internet era and evaluate some existing identity management frameworks in the context of IoT.

2.3.1 The Law of Identity in the Philosophy of Logic

Identity was firstly formalized by Aristotle's Law of Identity in logic as: "each thing is identical with itself." Coupled with the Law of Contradiction and Law of Excluded Middle, the conclusion that identity is an equivalence relation with the characteristics of reflexive, symmetric and transitive, could be drawn [DO 07]. Later, Wilhelm Gottfried Leibniz formulated Leibniz's Law, namely Identity of Indiscernibles [For96] as: "No two objects have exactly the same properties." Consequently, the two following principles derived from the Leibniz's Law, the Indiscernibility of Identicals) (Principle 1) and the Identity of Indiscernibles (Principle 2) are used for distinguishing two different individuals in the physical world and the cyberspace of the Internet due to the intuitive and straightforward recognition.

Principle 1 *For any x and y, if x is identical to y, then x and y have all the same properties.*

$$\forall x \, \forall y \, [x = y \rightarrow \forall P(Px \leftrightarrow Py)] \tag{2.1}$$

Principle 2 *For any x and y, if x and y have all the same properties, then x is identical to y.*

$$\forall x \, \forall y \, [\forall P(Px \leftrightarrow Py) \rightarrow x = y] \tag{2.2}$$

However, there are many paradoxes of the previous formalized definition. The most well-known paradox is the Ship of Theseus [DO 07], where people cannot tell whether the two wooden ships are the same one with replacing and reassembling of the planks and beams. In details, suppose all the planks and beams have been replaced through continuous repairing work over time. From the perspective of Principle 1, if the new ship (Ship B) is identical to the old ship (Ship A), they should have the same properties (planks and beams). However, the truth is that they do not have the same planks and beams. Another unexpected situation is: if the replaced planks and beams are reassembled into Ship C, we can barely say, the previous Ship A and current Ship C are the same ones even if they are composed by the same planks and beams according to Principle 2. In other words, only relying on attributes is not able to identify an entity in the IoT era where all individuals, collectives and things are interconnected. Unfortunately, many identity management solutions in the cyberspace, are based on the Leibniz's Law, where individuals are identified using a set of attributes and authenticated using credentials such as passwords. Due to this definition, systems, especially financial service providers, need to follow the Know-Your-Customer (KYC) [Hod02] procedure to verify the identity and store identities of its clients, which gives rise to fragmented identities and renders personal identity information theft even more severe. Because, in the Leibniz's Law (attribute) based identity model, users cannot prove their identities unless providing sufficient sensitive personal information and every system becomes a stand-alone identity store (also known as an Identity Silo). Personal identity information is over-harvested by many untrusted service providers and hence easily stolen due to inadequate defense measures of some unreliable systems.

2.3.2 Traditional IdM Models in the Internet

Digital Identity Management (IdM) systems are responsible for managing users' identity information, consists of identifiers (UserID, Email, URL, ...), credentials (Certificates, Tokens, Biometrics, ...) and attributes (Roles, Positions, Privileges, ...) [ITU09]. Since the birth of Information Technology, IdM systems have always been regarded as the keystone to access services and resources on the Internet. Over the past three decades, IdM systems have evolved from isolated to centralized and then to federated models [JP05].

In the isolated IdM model, identity providers have played central roles as relying parties (service providers) by providing subjects (users) accesses to Internet services and resources maintained by a single security domain [JP05]. When subjects decide to access Internet services, the first step is to register themselves to service providers and obtain digital identities with credentials from their security domain. Nevertheless, the rapidly proliferating of online services in various security domains incur the identity bloating. It becomes a mission impossible for a human to manage many digital identities (e.g., memorizing their corresponding passwords) by following the isolated IdM model.

In order to handle this problem, the centralized IdM model was designed through detaching identity management from service provision and allowing several service providers to rely on the same identity provider[JZS07]. Albeit, the centralized IdM model reduces the number of user identities, different security domains divided by different centralized IdM providers are not able to communicate. Consequently, the federated IdM model attempts to establish trust relationships between identity providers by which it becomes possible for users in one security domain to access services from another domain [MR08]. For example, Shibboleth [CS05], a federated identity management system for the Web, allows users to sign in to a security domain using just one identity and grant access to various systems belonging to the same federation of different organizations or institutions. The federated identity allows the sharing of information about users from one security domain to the other domains in the federation, which means that no matter which identity is authenticated in one domain, services provided by another domain in the same federation are accessible based on credentials provided by its domain. However, the access of many unauthenticated third-party service providers to the detached identity providers could also cause the spread of phishing attacks.

The emergence of centralized and federated IdM systems indeed alleviates the complexity of managing many identities originated from different security domains. However, the increasing number of applications per domain renders all agreements, protocols, standards, and processes (i.e., authentication and authorization) across these domains extremely complicated and undermines the usability of identity. Besides, centralized and federated IdM systems are designed from the perspective of service providers [AHS11], they are still not flexible for lacking users' consideration.

User-centric identity management models have been proposed to improve user experiences and ensure security and privacy [Ang+10]. Jøsang and Pope [JP05], for example, propose a user-centric IdM system where users manage their identities from different domains via personal trusted devices like smart cards or phones,

which means that all user identities are maintained by few personal devices. The OpenID [RR06] is also a user-centric and decentralized identity system for Web services. It introduces ID token (JSON Web Token) based on the OAuth 2.0 [Har12] authorization protocol to authenticate users. The decentralized framework makes the identity providers more robust for resisting distributed DoS attacks. However, the identity providers taking OpenID standard could see all the related web login information which also makes cross-site tracking easier. Besides, the URL-based identifiers in OpenID usually compromise users' privacy. Suriadi et al. [SFJ09] propose an identity management system where they integrated federated Single Sign-On (SSO) and follow user-centric design principles to build identity management, taking into account user privacy. In [AKS09], Ahn et al. enhance the privacy functionality in user-centric identity systems via applying privacy labels to personal claims so that the privacy of users could be protected according to different secrecy level. Although the user-centric identity management systems provide improved solutions to manage identities of subjects and service providers, the trust assumption that users still have to put all the trust on the third-party identity providers is still in there and has not been eradicated. Users still have to rely on the "trusted third parties" identity providers to access services in different domains while these identity providers could see all the transactions between users and service providers.

2.3.3 Existing IdM Systems in IoT

In this section, we identify several initiatives and groups working on digital identity management from the traditional Internet including PRIMELife [PRI11], SWIFT [Pér+11], DAIDALOS [DAI03], Kantara [Kan09] (Liberty [Lib01]), FIDIS [Mat+09], SAML [Arm+08], Higgins [Gui08], OpenID [RR06], Shibboleth [CS05], STORK [STO08], PICOS [PIC07] and Cardspace [BSB07]. Then, we evaluate these existing IdM frameworks under the context of IoT in Table 2.7. Specifically, we need to consider the previous unconventional characteristics, security and privacy requirements in that these are of great importance with an impact on the design of identity management framework for the IoT:

- **Scalability**: Internet of Things will comprise billions of individuals, collectives and everything in the cyber-physical world, which demands highly scalable identity management. Using the traditional centralized IdM scheme where all IoT identities are maintained by one universal third party to builds the highly scalable IdM solution becomes extremely unrealistic. Inevitably, there will be many different identity providers from different IdM systems. Albeit, federated identity management solutions like SAML [Arm+08] and Shibboleth [CS05] where different identities from different IdM systems could be managed, break the barriers between different IdM systems following the federation standards and successfully bring a silver lining of designing the IoT IdM systems. However, in trustless networks trust should be taken into account, that is how to build mutual trust relationships between different IdM systems. Consequently, identity management should be scalable and trusted in distributed trustless networks without centralized

control of any security authorities (i.e., identity providers, central access servers).

- **Interoperability**: The broad spectrum of connected objects makes them extremely heterogeneous with different communication, information, and processing capabilities. Each connected object would be subjected to various technologies such as wireless communication technologies (IEEE 802.15.4, WiFi, Bluetooth Low Energy, ...), communication protocols (CoAP, LORA, MQTT, ...), cellular communication technologies (GSM, UMTS, LTE, ...) and hardware-dependent controllers (Arduino, Raspberry, Eaglebone,...). Diversity and heterogeneity lead to interoperability problems; therefore, unifying all identities of IoT objects from different manufacturers, vendors, communities and standard groups, has been considered to be a mission impossible. As IoT continues to evolve, emerging standards (e.g., OneM2M, IoT reference architecture) remains highly fragmented in terms of vocabularies, methods, and models. The design of "interoperable" identity management of IoT objects remains balkanized without an integrated approach to make substantial progress in reducing software, hardware, and communication heterogeneity.
- **Mobility**: The IoT is ubiquitous which means IoT devices, such as vehicles or wearable devices are subject to strong mobile capability. No matter where these devices are located, we need to authenticate ourselves, get authorization and access controls to the corresponding device services. Therefore, identity management for the IoT should be characterized by mobility and requires peer-to-peer authentication and authorization services.

In addition to these IoT characteristics, the design of IdM is also confronted with challenges from the **security** perspective. For instance, most of the IdM solutions are under the assumption that the subjects and relying parties trust the IdP, which increases threats from internal attacks of the IdM and hence compromises security services like confidentiality and integrity. Besides, the IdM should be robust enough to defend some vulnerabilities and longstanding security attacks such as single-point failure or phishing [AHS11], which undermines the availability of IdM. Therefore, eliminating the trusted third party and building a trusted IdM in trustless networks are pretty critical to the distributed IoT environments. IdM systems as intermediators are responsible for dealing with all transactions between Subjects and Relying Parties (RPs). Consequently, the IoT IdM should be equipped with capabilities of providing anonymity or pseudonymity to RPs in order to preserve **privacy**.

From Table 2.7, we can see, admittedly, many initiatives attempt to develop user-centric identity management systems but they still not entirely satisfying IoT requirements. Users have to consider all entities from IoT and coordinate different application domains to join the IdM system, which compromises scalability and increases the difficulty of building interoperable IdM systems in such heterogeneous environments. Although some initiatives like OpenID or PICOS are capable of extensibility due to their decentralized architectures to some extent. Nevertheless, they cannot deal with mobility in networks of ubiquitous IoT devices or services. No matter where devices are and where they move, the mobile IdM system should ensure the usability of users' identities. At last, most of the

Table 2.7 – Identity management initiatives comparison

	Scalability	Interoperability	Mobility	Security & Privacy	User-centric
PRIMELife(PRIME)			★	★	★
SWIFT(DAIDALOS)		★	★	★	★
Kantara(Liberty)	★	★	★	★	
FIDIS		★	★	★	★
SAML		★			
Higgins		★		★	★
OpenID	★				★
Shibboleth				★	
STORK		★		★	★
PICOS	★	★	★	★	★
Cardspace		★	★	★	★

initiatives handle security and privacy aspects under the common assumption that all users including subjects and relying parties should trust their IdPs, which are involved in every transaction, and consequently undermines user privacy. Similarly, despite adopting the user-centric model, these identity management systems still have to rely on third-party identity providers, which does not eliminate the privacy concerns.

To sum up, current Leibniz's Law (attribute) based identity management systems on the Internet, have evolved from the isolated model to decentralized, federated and user-centric identity management models such as OpenID adopted by many online service providers (i.e., Facebook and Google). Usually, the IdMS of some big online service providers becomes the universal identity provider in their federated domains. The federated identity solution establishes some certain relationship among online service providers. Therefore, we can log in other online services only using one account such as our Facebook or Google account. Admittedly, the federated user-centric IdMS using relationships indeed alleviates the complexity of users managing their identities. However, security and privacy have not been solved correctly in that users have to put all the trust on their identity providers who are sitting in the middle and can see all activities between every user and their online service providers. Therefore, eliminating the unnecessary third parties and building a trustless identity provider in peer-to-peer networks are pretty critical for IoT infrastructure with user-centric security.

2.4 State of the Art of Access Controls

Access control refers to a security mechanism which regulates who can access what kind of resources or services in computer systems. The development of information systems since the 1970s has given rise to various access control models in order to handle increasing application requirements. At early days, the access control matrix [But74] was introduced to describe the security state in computer systems formally. Rows in the access control matrix prescribe access permissions to all programs, data, and resources according to different users while columns set access permissions to users according to resources. However, it is challenging to manage

a large access control matrix when we have many users and resources (e.g., 1000 users * 1000 resources = 1 million entries). Later, access control lists (storing policies by column) and capability-based access control (storing policies by row) models are created to boost efficiency by deleting additional access control entries.

- In access control lists (ACLs), resources store access lists regulating granted users who can access the resources.
- In the capability-based access control (CAC) model, users have their capability lists which prescribe all available resources.

Other access control models such as role-based access control (RAC) and attribute-based access control (ABAC) models, are improved management over users and resources. For instance, the RAC model is designed to manage users efficiently. Through assigning roles to users, administrators can easily manipulate the process of granting access permissions instead of granting permissions to each user. The ABAC model [BPS16] is born to meet requirements of intensive management over resources in terms of multi-factors such as location, time, battery life. With the advent of IoT era, many traditional access control models such as the ACLs and RAC models [San+96], which are designed for centralized systems, become obsolete due to the rapid growth of roles and policies. Besides, more and more factors and parameters such as time or location should also be taken into consideration in designing access control solutions. Albeit, the ABAC model aims at handling this problem, the existence of centralized identity providers in the ABAC model still has to face up to the scalability issue. A common problem of existing solutions stems from centralized administrative parties (i.e., administrators or identity providers) that become indispensable for assigning access rights, roles and attributes, and, consequently, these solutions are not suitable for scalable decentralized IoT systems. The CAC model [Her+13; GPR13; HBF17] raises much attention by virtue of its flexibility. However, it has the same premise by which users who request services need to rely on the authentication of third parties like identity providers or certificate authorities. This assumption is unsuitable in trustless IoT environments where user accounts could be generated by each subject (e.g., human) without the endorsement of other intermediate parties. In other words, the CAC model only works in trusted environments based on federated and decentralized identity management systems. Therefore, none of the proposed access control models (ACLs, RAC, ABAC or CAC) can satisfy the scalable, interoperable and trustless IoT environments. The access control mechanism for IoT should be rebuilt based on IoT characteristics, security and privacy premise.

Over the past decade, the rise of Web 2.0 applications, especially social network systems such as Facebook, Twitter, or Instagram, has greatly facilitated the study of the relationship-based access control (RBAC) model [Fon11; Bru+12; Akt+13; CPS14; APS16; ASP17]. Similarly, the RBAC model is another refined management over users, in which owners of data (e.g., photos on social networks) take advantage of relationships with others to manage access permissions. Some features of social networks (e.g., anyone can freely build or join social networks) make the RBAC suitable for open and distributed systems. Besides, owners, as the only responsible party for their access control policies, can grant access permissions based on the relationship between them without relying on administrators

or third parties. Furthermore, the RBAC model can easily integrate other access control models (e.g., RAC, ABAC). For instance, Cheng et al. [CPS14] integrate attribute-based policies into the RBAC model to meet various security and privacy requirements in online social networks. However, the current RBAC model is restricted to closed online social network platforms such as Facebook, Twitter, or Instagram. Although owners could set their access policies by different trust relationship, owners cannot fully control their data or asset in that these companies are the actual controllers of users' data.

In addition to access control models defined at design-time, access control mechanisms of dynamically defending threats and attacks at runtime should also be taken into consideration to ensure the availability of services in IoT systems. Current cyber-security solutions are far from being satisfactory to deal with the exponential growth in the number and complexity of cyber-attacks [DYW11]. Besides, efforts and knowledge required to launch sophisticated attacks are decreasing while their propagation has been reduced from days in the early 80s to a fraction of seconds in 2000s. In order to detect security attacks, there are two basic intrusion detection techniques: signature-based intrusion detection systems and anomaly-based intrusion detection systems [CS15]. Signature-based Intrusion Detection Systems (IDS) build a database of known attack signatures or identities. However, these systems cannot detect new types of attacks or even a known attack with a slight change in its signature. The main feature of the anomaly detection approaches is their capability in detecting novel and new attacks. The anomaly-based IDS define a baseline model for normal behavior of the system through off-line training and consider any activity which lies outside of this normal model as an anomaly [FHN07]. Any attack, misconfiguration or misuse will lead to deviation from the normal behavior; we name it as abnormal behavior. The main limitation of this approach is a large number of false alarms that can be produced.

In summary, the proposed ACLs, RAC, ABAC, CAC, and online social network RBAC models do not work in trustless environments due to relying on centralized administrative parties (e.g., administrators, identity providers, operating companies of social networks). Furthermore, eliminating these unnecessary third parties and building a trustless identity provider in peer-to-peer networks are pretty critical for IoT infrastructure with user-centric security. Therefore, a new access control framework for trustless environments needs to be proposed to manage access permissions and ensure the availability of IoT systems through actively defending various attacks.

2.5 Building IoT Secure Solutions: Principles and Approaches

According to the design requirements, we have listed the principles and approaches in Table 1.2, which are used to build our secure IoT infrastructure. In this section, we dive into these approaches, namely, the blockchain technology, social IoT, service-oriented architecture, asynchronous and non-blocking processing architectures, to find out the reasons why we use these approaches and principles,

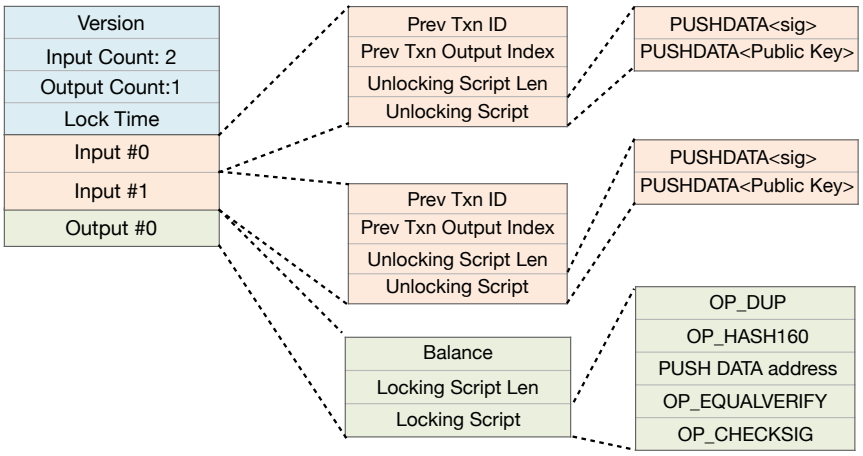


Figure 2.1 – Bitcoin transaction structure

and to see the state of the art of applying these approaches and principles to IoT systems and infrastructures.

2.5.1 Distributed Model and Blockchain Technology

From existing IoT solutions, distributed model [RZL13] could not only improve the scalability of the IoT systems but ensure security and privacy of users compared with centralized and decentralized models in that users do not have to trust Cloud service providers or third-party identity providers and regain the full access right to their data and identity information. Webinos [Fuh13] and Alljoyn [All16] are pioneers for practicing distributed model in IoT environments and hence harvest scalability and security advantages. However, these distributed systems cannot solve the non-repudiation identity problem since, in theory, anyone can setup a so-called "trusted" identity provider. Until the blockchain technology emerges as a prominent perspective to develop IoT security solutions in decentralized and trustless environments [HP16; Bas17; ABM17]. By using blockchains, we could remove the intermediaries, and allow users and devices to manage their identities without relying on third parties or intermediaries.

The blockchain technology, which is initially introduced by Satoshi Nakamoto in [Nak08], keeps permanent records of all transactions used to transfer bitcoin values between members, participating in the Bitcoin peer-to-peer network. In order to realize the decentralized settlement without the centralized intermediaries like banks, Nakamoto firstly defines the structure of transactions. As shown in Fig 2.1, the structure of a transaction in Bitcoin could be divided into four fields: version, inputs, outputs, and locktime. The outputs and inputs record how could the bitcoin values transfer between private key owners. Specifically, a locking script in the output field defines a spending condition while an unlocking script in the input field specifies a solution that satisfies the spending condition in the previous unspent transaction output (UTXO) to finish the value transferring. As the key concept of the Bitcoin system, transactions are basic units of transferring the bitcoin value. Almost all of the operations in Bitcoin are relevant to transactions so that transactions can be created, broadcast, propagated, validated, and finally added to the transaction ledger [Ant14]. Then, relying on the structure of

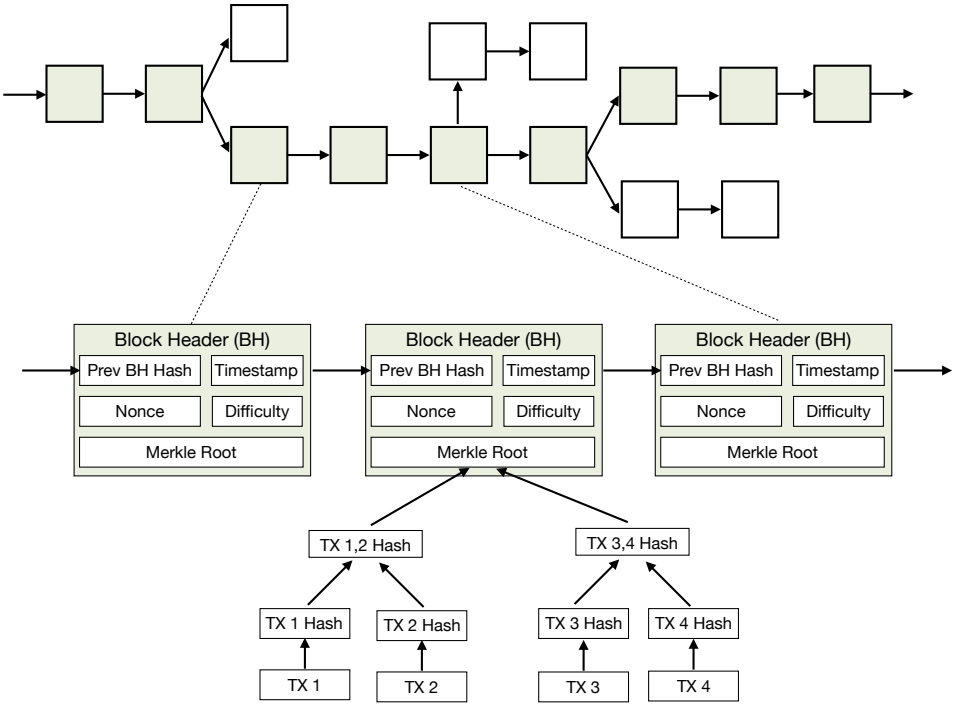


Figure 2.2 – Blockchain representation in Bitcoin

Bitcoin transactions, Nakamoto lays down a set of rules of sealing transactions into the chained blocks from four aspects:

- Participating full nodes could independently verify transactions;
- Mining nodes could independently seal transactions into blocks through the Proof-of-Work algorithm;
- Participating nodes could independently verify new blocks;
- Participating nodes could independently select the greatest-cumulative-work chain.

Through these rules, transactions could be sealed into blocks which finally will be appended by mining nodes. By such, decentralized settlement is achieved without the centralized intermediary banks. From Figure 2.2, we can see how transactions are grouped into blocks via the Merkle binary hash tree, how these blocks are chained together, and how states of blockchain reach consensus through the greatest-cumulative-work chain rule. Mining is the mechanism that allows the Bitcoin blockchain to be decentralized and secure without any central authority. Nakamoto introduces the concept of Proof of Work (PoW) as a mining process to ensure consistency of transactions and solve the double spending problem in decentralized Bitcoin networks [Nak08]. With the PoW, there is no need for any trusted authority, such as a bank, to keep track of the money transfer, all members have their tamper-proof copy of the blockchain ledger. Each node in the Bitcoin peer-to-peer network maintains a copy of the blockchain. Besides, the blockchain is simultaneously updated through the peer-to-peer network so all members can validate any transaction instantly.

Since 2014, Blockchain entered the 2.0 era leading by Ethereum [Woo14], which is a decentralized platform based on the blockchain technology. It aims at

creating a general purpose decentralized computer via the Turing-completeness smart contract concept, which allows writing and deploying all kinds of decentralized applications (Dapps) without any possibility of downtime, censorship or fraud. Vitalik Buterin [But14] explained Ethereum as: combining the cryptographic algorithms with the economic incentives to create a decentralized network with memory. To sum up, the blockchain provides us a new perspective to reconstruct the Internet in distributed P2P networks without any unnecessary intermediaries.

It is the concept of eliminating central authorities and intermediaries through blockchain technology that opens opportunities to multiple initiatives and research topics in the context of IoT security including identity and access control management[Bas17; OAA16; Dor+17]. Therefore, we surveyed to see how the blockchain could transform the traditional identity and access control management systems.

2.5.1.1 Elucidation of Identity and Naming Systems

The distinction between identity management systems and naming systems (i.e., Domain Name Service-DNS, Active directory or URLs) is blurry in the context of the Internet. At first sight, there is a slight difference between these types of systems. Identity management systems are coupled with service providers and used to identify resources or users in a particular domain whereas the naming systems are designed to identify computers across networks, resources and user accounts in companies or social networks. DNS records, URLs and user accounts are somehow identities at the same time.

Therefore, at first, researchers used blockchains to build identifier or naming systems since 2014. The Namecoin [Nam14], for example, is a fork of the Bitcoin blockchain that provides domain naming functionalities via binding human-readable names and IP addresses. It is the first solution for naming trilemma of the Zooko's Triangle [Zoo18] on building a secure, decentralized and human-meaningful naming system. By modifying Namecoin, Certcoin[FVY14b] builds a decentralized authentication system (PKI), which defines a set of key operations like registering, updating, verifying and revoking. Based on Certcoin, Authcoin[Lei+16] proposes a new alternative protocol for authentication using a flexible challenge-response schema to PGP in the context of the Web of Things. Its successor, the Blockstack[Ali+16] attempts to redesign the naming system and PKI authentication features using state machines. It also adds the storage aspect to its blockchain-based system in order to construct a new type of Internet resource identification, preserving privacy and including property rights. Fromknecht et al. [FVY14a] have tuned the certcoin parameters to ensure the retention of identities where users could not register the same already-registered identity anymore.

Even though naming systems could be exploited as identity providers for individuals in specific domains, the goal of identity systems and naming systems is completely different. The former attempts to find a way to uniquely define individuals in the cyberspace whereas the latter is responsible for routing by assigning a unique identifier to retrieve the user or object in the service domain. Instead of working on naming systems, many research teams and recent projects

Table 2.8 – Blockchain based identity management systems comparison

	DNS	PKI	Storage	Bitcoin based	Ethereum based	Full stack	Reputation	Privacy	Year
Namecoin	★			★					2014
Certcoin	★	★		★					2014
Fromknecht	★	★		★					2014
Uport		★			★				2015
Sovrin		★				★	★	★	2016
Jolocom		★			★				2016
Blockstack	★	★	★	★					2016
Authcoin	★	★		★					2016
ChainAnchor		★				★		★	2016
Liu et al		★			★		★		2017
NEXTLEAP		★				★		★	2017
Azouvi		★			★			★	2017
Axon		★		★				★	2017
Augot		★		★				★	2017
SCPKI		★			★				2017
BlockID		★				★			2018
BIDaaS		★				★			2018
Borse et al		★			★			★	2019

as introduced in the following section, are working on the digital identity problem based on blockchains. These blockchain based identity frameworks are promising of becoming the critical component of future digital world infrastructures.

2.5.1.2 Blockchain-based IdM Systems

From an academic research perspective, Blockchain-based IdM systems are gaining much attention to propose new solutions for digital identities: Bassam [Bas17] introduces a blockchain-based PKI and implements the identity management system based on Ethereum smart contracts. In his work, he defines several identity-related operations like adding attributes, signing attributes and revoking signatures. More importantly, he calculates the cost of different operations in the Ethereum platform. Liu et al.[Liu+17] develop an identity management system based on Ethereum smart contracts through binding public key and user's entity information. Besides the identity management part, they also redefine the token to fit their proposed reputation model to reflect the reputation of users. Axon[Axo15] analyzes privacy requirements when designing decentralized PKI systems and proposes a blockchain-based PKI with privacy awareness. In addition to a set of operations like registering, revocation and recovery, they introduce the concept of the neighbor group to enhance the performance of privacy preserving. Augot et al. [Aug+17b; Aug+17a] modify the Bitcoin stack to build an identity management system and introduce a zero-knowledge proof called Brands selective disclosure scheme [Bra00] to ensure the anonymity of the identity at the same time. Hardjono [HP16] introduces a blockchain-based and privacy-preserving identity solution called ChainAnchor using zero-knowledge proof in a permissioned blockchain environment. In ChainAnchor, verified nodes have the privileges to write or process transactions, and others could only read and ver-

ify transactions. All verified nodes are built on tamper-resistant hardware and form the privacy-preserving layer to provide privacy protection services to users. Halpin [Hal17] introduces NEXTLEAP, a federated identity system with privacy-preserving features using blind signatures. Moreover, they make use of authentication services to build a more secure messaging application. Azouvi et al. [ABM17] also propose a privacy-preserving identity solution using blind signatures. They set up a threat model, do the security analysis and implement their solution in Ethereum. Gao et al. [Gao+18] present a blockchain-based identity management framework for people called BlockID, into which integrates biometric-based user authentication and trusted mobile computing technology. Lee [Lee18] introduces a digital identity management framework called Blockchain based Identity as a Service (BIDaaS) and gives a practical example of mobile users. Faber et al. [Fab+19] also propose a Blockchain-based Personal Data and Identity Management System (BPDIMS), complying with the 2018 EU General Data Protection Regulation (GDPR). BPDIMS defines vital stakeholders, including users, service provider, data purchaser, and data validator, to sort out responsibility, power, and interest in personal data management. Borse et al. [Bor+19] design a privacy-preserving identity management scheme based on the Ethereum smart contract. It incorporates zero-knowledge proofs and cryptographic commitment techniques to provide disclosure of users' identities selectively.

In addition, several startups and IT players are focusing on the development of identity systems such as Uport [Upo] , Shocard [Sho], Bitnation [BIT], Civic [Civ], Jolocom [Jol], Sovrin[Sov], Evernym [Eve], ID2020 [ID2], to mention a few. For example, Uport, which is a core component of the Consensys Ethereum ecosystem [Con], aims at building decentralized applications to solve the digital identity problem. It mainly uses smart contract to design digital identity model and ensures reliability and usability of identities through a set of operations (i.e., keys revocation and identities recovery). Sovrin takes a different approach and provides a complete full stack to manage identities from the distributed ledger to devices. It adds the identity layer for every entity on the Internet and operates as a global public utility designed to provide permanent, private and trustworthy identities. Sovrin establishes a public permissioned blockchain in a peer-to-peer network in which nodes are divided into authenticated validator nodes (permissioned) and observer nodes to ensure high performance and scalability.

Table 2.8 summarizes and compares some blockchain based identity management frameworks from our previous mentioned solutions. In general, the blockchain-based identity is also called the self-sovereign identity, which denotes an approach that transferring access control rights and management of identities from traditional identity providers to the edge under the control of identity owners. Only owners have the right to dispose of their identities, which prevents attacks from malicious third-party identity providers. Although there are two different methodologies, permissioned (e.g., Sovrin) and permissionless (e.g., Uport), to implement the blockchain self-sovereign identities, the basic concepts could be summarized as:

- Identities of individuals (i.e., human beings) and collectives (e.g., companies, banks, governments) can be selectively stored in the blockchain without compromising privacy.

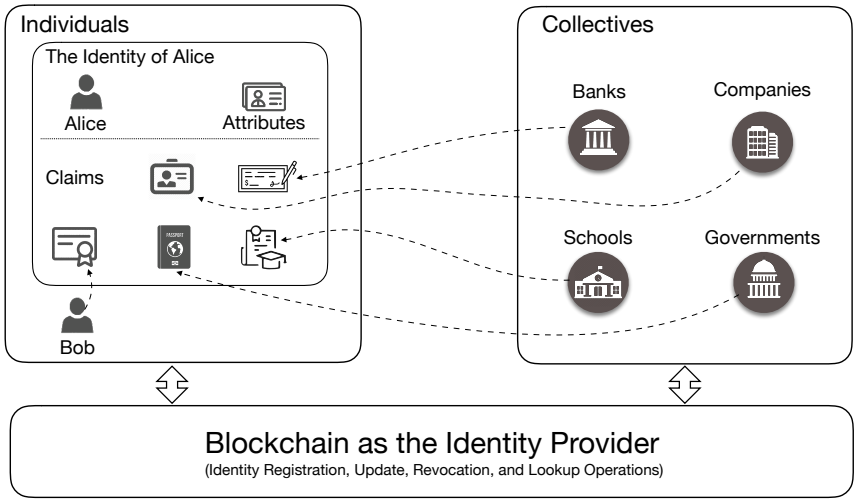


Figure 2.3 – Overview of blockchain based identity management systems

- Individuals and collectives issue claims to each other using these blockchain identities. Generally, claims are the endorsement by other individuals or collectives, which could be, for example, governments, banks, universities or even friends.

As shown in Figure 2.3, there are two individuals (Alice and Bob) and several collective examples (i.e., companies, banks, governments, and schools). The individual or collective identities, composed of the identity attributes and identity claims, could be gradually completed through the following steps:

- Individuals or collectives, for instance, Alice, could generate and add as many identity attributes (e.g., identifier, public and private key pairs, biometrics) as required.
- Individuals or collectives will create blockchain identities by submitting identity-related information such as public keys and corresponding signatures.
- Individuals or collectives could use public and private key pairs, which are correlated to the mined blockchain identity, to issue claims.

In Figure 2.3, Alice could self-generate identity attributes and also receive the claims from her employers, banks, government, schools and even her friend Bob. In different scenarios, Alice could give the necessary identity claims to identify herself or to demonstrate that she has some qualifications. For example, when applying for a job, Alice can give her ID Card from the government and diploma from her university. After entering the company, she has to give her bank account to receive the salary.

The self-sovereign blockchain based identity management systems eliminate unnecessary centralized identity providers by creating the blockchain identity on the blockchain platform, in which all users and service providers follow the identity consensus and hence could verify identities instead of blindly trusting in some big third-party identity providers. Moreover, the concept of claims in the blockchain based IdMS essentially is an extension of relationships in the federated identity management model. Claims, as the endorsement relation from others, are indispensable in the trustless distributed blockchain based IdMS, since individuals still

do not trust (or know) each other, even if they could verify the real identities with privacy concerns. However, claims are restricted in endorsements from others, which cannot be used to express bidirectional relationships between users. More importantly, claims lack of extensible consideration for access controls, which are a critical part for building IoT secure infrastructures.

2.5.1.3 Blockchain-based Access Controls

Recently, some researchers try to use blockchain technology to design access control systems for IoT [OAA16; OHL17; MMR17]. For instance, Ouaddah et al. [OAA16] present FairAccess, a blockchain based access control framework, which makes use of bitcoin stack and introduces a new type of transactions to manage access permissions. They rely on blockchain transactions to grant or deny access requests in IoT environments through defining the access control transactions using smart contracts, which takes advantage of the immutability of the blockchain and makes the auditing of access control policies transparent. Blockchain-based access controls eliminate the need for centralized authorities to set access control policies in constrained IoT devices. They rely on blockchain transaction data models in order to execute access control operations to grant or deny access requests in IoT environments. Another benefit is the auditing of access control policies, which can report immutable transactions in the blockchain. However, the blockchain is not a panacea for all kinds of problems. Since the storage resources are pretty scarce and expensive in public blockchains, it is hard to imagine that access control policies of all IoT devices are uploaded to these blockchains.

Although the private blockchain access control mechanisms could solve the storage consumption problem, the isolated private blockchain systems will undoubtedly hinder the large-scale adoption due to the interoperability issue. Besides, blockchain based access controls also raise privacy concerns in that all transactions are publicly mined into blocks, especially bitcoin or ethereum based access control solutions [OHL17; MMR17].

In summary, blockchains of eliminating central authorities and intermediaries open opportunities to multiple initiatives and research topics in the context of IoT. The self-sovereign blockchain based identity management systems can eliminate unnecessary identity providers by creating blockchain identities on the blockchain platform. By such, users can verify identities from trustless blockchain based identity providers instead of putting all trust on their third-party identity providers who are sitting in the middle and see all activities between every user and their online service providers. However, claims are restricted in endorsements from others, which cannot be used to express bidirectional relationships between users. More importantly, claims lack of extensible consideration for access controls, which are indispensable for building IoT secure infrastructures. Also, the blockchain based access control frameworks are not viable for building our IoT secure infrastructure when it comes to expensive on-chain storage cost and privacy concerns.

2.5.2 Socialized IoT

Generally speaking, the goal of having an identity is for facilitating connections or communications among entities (e.g., human). As a result, where there is no interaction (social relations), there is no identity. A lone wolf does not need to rely on the identity too much in that living independently instead of with a group does not need to interact or communicate with others socially. From the human evolution perspective, the more complex social relations become, the more difficulties humans could tackle. A social network is composed of social entities (such as individuals or organizations) and social relationships between these social entities. The social attributes, which are inherently existing in human society, are indispensable for human beings. This law applies equally to the cyberspace, which is composed of IoT entities.

Recently, researchers from the academic community have started to apply the social network principles to IoT forming the Social Internet of Things (SIoT) [AIM14]. Atzori et al. [Atz+12] integrate social network concepts into IoT through establishing social relationships management model, in which relationships are classified into several categories such as parental, co-location, co-work, ownership, and social objects. More importantly, they propose an implementation architecture of their proposed SIoT system, which consists of relationship management, trust management, service discovery, and service composition. Based on the SIoT model proposed in [Atz+12], Nitti et al. [NGA14; Nit+17] present trust management models according to behaviors of things in order to build a reliable SIoT system. Farris et al. [Far+15] take advantage of virtualization concepts to build virtual representations of things, which is beneficial to improve visibility and enhance interoperability of things in SIoT systems. Girau et al. [GMA16] introduce objects discovery algorithms for SIoT via communication channel scanning or localization features of objects, which improves visibility of objects and services. The SIoT paradigm seeks a transition from isolated devices to friendly unified devices which could find friend devices and manage their relationships. By such, devices can be involved in social-like networks, in which devices can publish their services to improve visibility and find services provided by other devices. With services discovery and composition, SIoT hence makes devices smarter and able to interact without human interventions. Besides, relationship-based trust management could be built according to the history of interactions between things. Therefore, applying social relationships to IoT has many advantages:

- relationships among social entities make it possible for all entities to join a social network and interact with each other. Social entities thus form a scalable and ubiquitous IoT network;
- relationship based services discovery and composition makes heterogeneous IoT interoperable;
- relationships enable the trust to accumulate as interactions increase.

To sum up, the socialized IoT paradigm does point out a new direction [Atz+12] to solve the previous IoT challenges such as scalability and heterogeneity due to the inherent characteristics of social networks such as extensibility, human-centric, and ubiquity. However, security aspects in SIoT remain unclear.

2.5.3 Distributed Computing with Service Oriented Architecture

As a software engineering architecture, the SOA was introduced by Gartner [Gar] in 1996, which aims at providing a more elastic and real-time responsive enterprise software architecture. However, it did not receive much attention until the Web-based services became widespread. Especially, with the development of various Web-based application frameworks, people began to realize that the Web service-oriented software architecture contributes to solving the interoperability issue between heterogeneous systems in distributed enterprise software architectures. After 2005, the SOA was gradually standardized by OASIS [OASa] and World Wide Web Consortium (W3C) [W3C] through three specification documents including the Service Component Architecture (SCA) [OASb], Service Data Objects (SDO) [IBM04] and Web Service Policy (WS-Policy) [W3C07], which indicates the entry of SOA into the implementation phase. The SCA and SDO form the basis of the SOA programming model whereas the WS-Policy sets out specifications of secure interactions between SOA components.

In general, the SOA could be defined as the composition of loosely coupled software components, which provide services to each other through agreed communication protocols such as the Enterprise Service Bus (ESB). Many features in the SOA, such as loosely coupled, reusable and composable, make components highly modular so as to deploy them according to practical requirements. In detail, SOA mainly involves a set of standards for describing every service component (e.g., the Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL)), service discovery, and service composition mechanisms. On the one hand, heterogeneous IoT devices, following the SOA, are interoperable through universally predefined application interfaces. On the other hand, service-oriented things can self-configure, self-adaptation and self-composite to handle dynamic changes in the context of IoT without human interventions.

Even in IoT environments, there are a large number of IoT solutions [De +08; ERA09; Car+09; RLN11] built using the concept of SOA, which are called Service Oriented IoT Middlewares (SOM). For instance, De Souza et al. [De +08] take advantage of the SOA to build Web-based services which can easily be integrated services of their physical devices into IT-systems. Adopting the SOA in the design of IoT middleware solutions could solve the interoperability problem via providing universal interfaces like REST APIs in the heterogeneous IoT environments since everything could be encapsulated into services. Besides, service composition provides developers with great flexibility to handle dynamic changes in IoT. Under the SOA, developers could follow the communication protocols and workflow instructions to independently develop loosely coupled, reusable and composable components for any scenario (e.g., developing a customized component for constrained-resource devices). For example, Gatouillat et al. [GBM18] introduce a self-adaptation framework to handle dynamic changes in IoT environments, which relies on quality-of-service properties to specify control objectives and automatically adjust composable services (controllers).

In summary, the heterogeneous IoT devices in distributed peer-to-peer networks, need to rely on the SOA, which not only can alleviate interoperability problem of heterogeneous things but open an opportunity to deal with dynamic

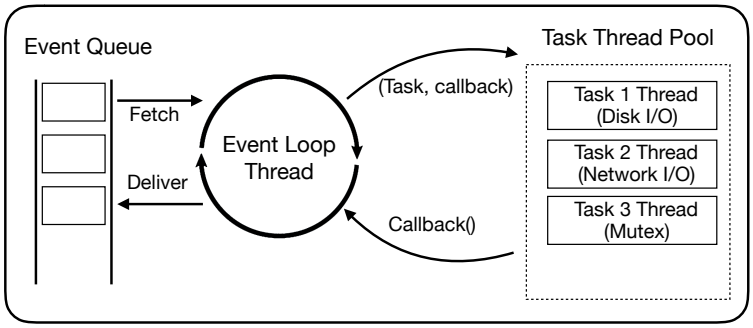


Figure 2.4 – Asynchronous and non-blocking representative model

changes in IoT environments. Similarly, when building IoT secure infrastructures, we can design the security functionalities such as identity management, authentication, and authorization as security services so that IoT systems are able to reuse and deploy them according to specific application scenarios dynamically.

2.5.4 Asynchronous and Non-blocking Processing Architectures

When developing application programs, the asynchronous and non-blocking processing means that 1) the application thread could continue to execute other processing (non-blocking) even if encountering the time-consuming operations such as network input/output, disk input/output, and mutual exclusion; 2) when the operation is finished, results will be delivered to the previous thread (asynchronous). As depicted in Figure 2.4, events in the event queue will be instantly fetched by the non-blocking event loop thread. If there are time-consuming tasks, the event loop will register callback functions, transfer tasks to other threads and continue to process the events. When the task is finished, the task thread will return the result through the registered callback function.

Since the dynamic changes of environmental parameters such as temperature, location or topology, pose challenges in real-time processing, the asynchronous and non-blocking processing architectures are essential to developing secure IoT systems. From Figure 2.4, we can see that the event loop is responsible for scheduling asynchronous operations and plays a critical role in asynchronous and non-blocking programming. Therefore, the event-driven model ensures the adaptability to dynamic changes. Besides, there are many applications of the asynchronous model like message-oriented model and publish/subscribe pattern, all of which are related to the event-driven model [Raz+16]. The message oriented model could be regarded as extended events through including senders and receivers to events while the publish/subscribe model is only another way of saying, from the perspective of event providers and consumers in the event-driven architecture. As the surveyed results are shown in previous Table 2.1, Table 2.3 and Table 2.5, most of the IoT systems have noticed the importance of the asynchronous and non-blocking processing architectures and integrate the support for the event-driven model.

2.6 Summary

In order to build our IoT secure infrastructure with user-centric security in peer-to-peer networks, we investigated research works and projects according to our criteria and evaluated them using established metrics of the unconventional characteristics and security requirements. According to our analysis on these selected IoT systems, we can observe that some principles or approaches such as the distributed and trustless frameworks, SOA, and event-driven model, are beneficial to handling IoT unconventional characteristics in IoT. Besides, trustless IoT secure infrastructures, specifically, trustless identity providers in distributed peer-to-peer networks, can eliminate trust to some unnecessary third parties, which significantly preserves privacy from users' perspective and brings user-centric security into a reality.

We also surveyed on identity and access control management frameworks which are keystones for building a user-centric IoT secure infrastructure. In the identity survey, we explored the origin of identity from the philosophy logic and cleaned up a comprehensive development roadmap of identity management systems on the Internet. Considering inherent IoT characteristics such as scalability, heterogeneity, mobility, security, and privacy, we identified the deficiencies of traditional Internet IdM systems in the context of IoT. Besides these unconventional characteristics in designing IoT solutions, the biggest challenge is *how to eliminate the unnecessary third-party identity providers to protect the security and privacy of users.* Therefore, we investigated self-sovereign blockchain based identity management systems which can eliminate unnecessary identity providers by creating blockchain identities on the blockchain platform. By such, users can verify identities from trustless blockchain based identity providers instead of putting all trust on their third-party identity providers who are sitting in the middle and see all activities between every user and their online service providers. However, claims in self-sovereign blockchain based identity management systems are restricted in endorsements from others, which cannot be used to express bidirectional relationships between users. More importantly, claims lack of extensible consideration for access controls, which are indispensable for building IoT secure infrastructures.

In the access control survey, we illustrated various access control models including Access Control Lists (ACLs), Role-based Access Control (RAC), Attribute-based Access Control (ABAC), Capability-based Access Control (CAC), and Relationship-based Access Control (RBAC). Then, we pointed out that *all of them do not work in trustless environments and need to rely on the authentication of third parties like identity providers, certificate authorities, or operating companies of social networks.* Besides these access control models in design time, we are experiencing grand challenges to secure and protect runtime IoT systems due to the significant increase in the attack surface. As a result, *a new access control framework for trustless environments needs to be proposed to manage access permissions and ensure the availability of IoT systems through actively defending various attacks.* At last, we investigated the SOA, asynchronous and non-blocking architectures, which could be adopted in designing the secure IoT solution to fit the heterogeneous, dynamic changes and limited resources IoT environments.

When building our IoT secure infrastructure in peer-to-peer networks, we firstly propose to use the blockchain technology to build a blockchain based IdP to eliminate unnecessary third-party identity providers. Then, we take advantage of the social relationships to design a universally social model for IoT, which unifies all IoT entities from people, companies, organizations to devices, applications, and services. The adoption of social concept in IoT and the global blockchain-based IdP make the IoT extensible and ubiquitous networks in which all heterogeneous social entities (namely, individuals, collectives, things) can freely join and leave based on created identities. More importantly, relationships from socialized IoT can be used to design an access control framework in trustless peer-to-peer networks. In detail, relationships between subjects and IoT things allow owners to set security policies to secure their own devices or services, which could integrate all IoT things to subjects with built-in security policies through ownership. We also follow the SOA, which could solve the interoperability problem by providing universal interfaces like the REST APIs in the heterogeneous IoT environments. Moreover, we take advantage of the asynchronous and non-blocking processing architectures (namely, event-driven model) in our implementation, which makes our IoT secure infrastructure deployed on edge nodes capable of handling high concurrency, real-time, and dynamic request in IoT environments.

CHAPTER 3

Secure Infrastructure for IoT

Contents

3.1 Introduction

In order to build an IoT secure infrastructure with user-centric security and take into account IoT unconventional characteristics, we make use of the blockchain technology, social IoT (relationships), SOA, asynchronous and non-blocking architectures to build blockchain-based identity management and decentralized IoT access control frameworks for peer-to-peer IoT networks. This chapter gives an overview of our proposed IoT secure infrastructure. Firstly, through our methodology, we elaborate our IoT reference model to point out which layer our proposed solution is located in and what kind of role our IoT secure infrastructure plays in the three layered-IoT reference model. Then, we recapitulate the approaches and technologies (blockchain, SIoT, SOA, asynchronous and non-blocking) for building our IoT secure infrastructure and state benefits of adopting them. After that, we give an overview of our proposed IoT secure infrastructure especially from the perspective of our blockchain-based identity management and decentralized IoT access control frameworks. At last , we describe a set of futuristic cases and scenarios to demonstrate the scalability and extensibility of our proposed secure infrastructure in IoT environments.

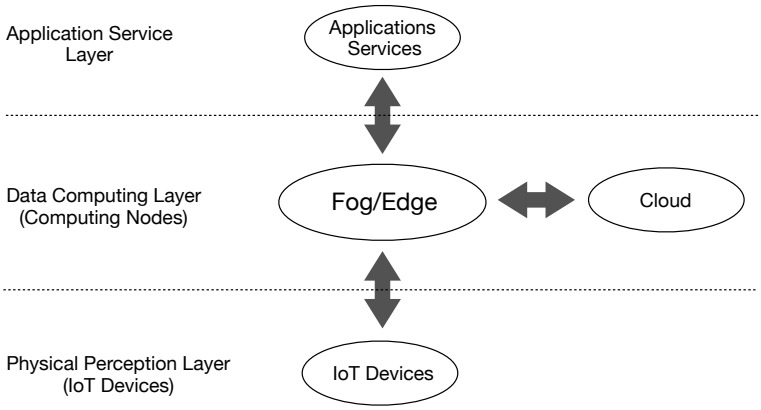


Figure 3.1 – IoT reference model

3.2 Methodology Outline

We define the IoT reference model as three layers: Physical Perception Layer, Data Computing Layer, and Application Service Layer as illustrated in Figure 3.1. The Physical Perception Layer (PPL) is composed of IoT devices which can sense and act on the environment. The Data Computing Layer (DCL) on the top of the PPL comprises two types of computing nodes: fog computing nodes and cloud computing nodes. The former are access points of IoT devices and do some lightweight data processing on behalf of users whereas the latter are responsible for the massive data mining work, whose data has been securely processed by fog nodes according to users' security policies in order to preserve privacy from users. Despite that the cloud computing has been considered as the panacea of processing and analyzing IoT data over the past decades, it shows drawbacks in many other aspects like latency, bandwidth, and mobility when transferring data from connected devices to the cloud. The fog computing paradigm provides elastic compute, storage and communication resources that take place in data hubs on smart mobile devices or the edge of the network in smart routers or other gateway devices. Due to a geographically distributed computing paradigm at the edge of IoT networks [Yi+15], fog computing provides low latency feedback, high bandwidth, and location-awareness services to the vicinity of IoT devices [Bon+12]. Moreover, the Application Service Layer (ASL) on the top of the DCL includes all designed application software providing services to users according to different application scenarios such as smart home, smart retail, smart healthcare.

To sum up, fog nodes are the core of our IoT reference model: fog nodes connecting all IoT devices, are the first computing units of IoT data from these devices; fog nodes communicating with the cloud, are indispensable especially in privacy-preserving; fog nodes directly providing application services to users, are usually trusted hardware that belongs to end users such as set-top boxes, cell-phones, or home routers. Therefore, combining the previous design principles and approaches (blockchain, SIoT, SOA, asynchronous and non-blocking), we build a secure infrastructure for IoT in the context of fog/edge computing paradigm, called Fog Computing Secure infrastructure for IoT (FoCuS), which relies on:

- The blockchain technology: which enables FoCuS with trustless, decentralized, immutable and public ledger that records users and organizations digital identities, and facilitates the development of new authentication and authorization mechanisms without relying on third-party identity providers.
- The social IoT: which allows FoCuS to establish relationships between social entities including individuals, collectives and things. Therefore, social entities and their relationships constitute social networks driven by humans. The socialized FoCuS makes the IoT extensible and ubiquitous networks where humans can register their identities, manage their devices and set consent, and access control policies according to different to relationship to share their device data.
- The SOA: by which we shape FoCuS into reusable software components that provides reusable security services, including identification services, relationship management services, authentication services, access control services, and threat detection services. These services are defined and developed through a set of universal REST APIs, which are loosely coupled, reusable and composable components for any scenario (i.e., developing a customized component for constrained-resource devices) in heterogeneous IoT environments.
- The asynchronous and non-blocking processing architectures : which enable FoCuS to handle high concurrency, real-time, and dynamic IoT environment based on the event-driven model.

3.3 Proposed Secure Infrastructure for IoT

As illustrated in Figure 3.2, our FoCuS consists of the blockchain-based identity provider and composable security services such as authentication, access control, relationship management, and threat detection. These services are grouped under the FoCuS infrastructure that supports service composition in order to build customized security services.

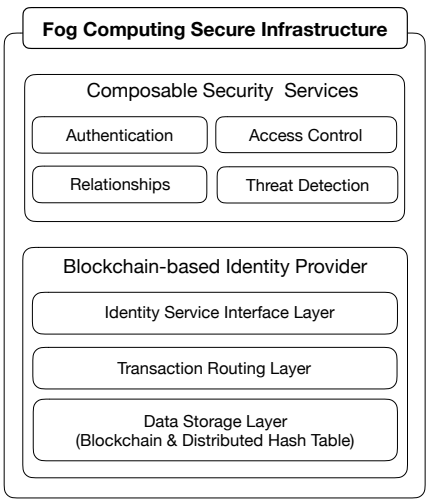


Figure 3.2 – SOA-based fog computing secure infrastructure

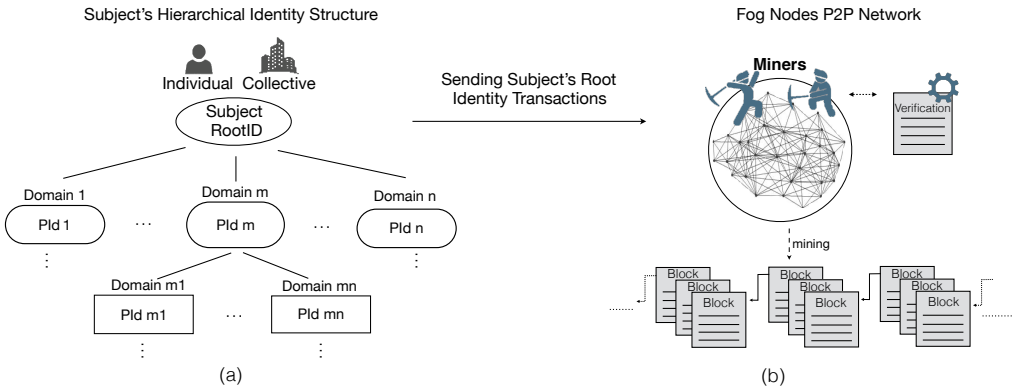


Figure 3.3 – Proposed fog computing paradigm (the identity management)

FoCuS makes all fog nodes interacting in peer-to-peer networks, which maintain blockchain identities where social relationships could be built between IoT entities such as people, devices and services. Based on identities and established relationships among social entities, composable security services such as authentication, access control, threat detection could be activated. FoCuS defines the parties in Figure 1.1 as:

- **Individuals or Collectives (Subjects)**. Individuals (i.e., human-beings) or collectives (e.g., companies, governments, organizations) are defined as subjects who possess root identities that can be used for blockchain identity transaction generation.
- **Things (Service Providers)**. In the IoT context, things refer to physical things (e.g., smart objects) and virtual things (e.g., applications, data, resources) [ITU12a]. In this dissertation, we use things to denote IoT services provided by software applications or IoT devices. Things are always correlated with their owners (namely, subjects). By such, subjects could obtain data by accessing services offered by IoT devices through ownership.
- **Ledger and Miners (Identity Provider)**. The identity ledger is an identity repository which records identity-related transactions of subjects. Some fog nodes play the role of miners which are responsible for writing identity to blockchain and reaching consensus on identity information.

As depicted in Figure 3.3, things in IoT are correlated to their owner (subjects) through hierarchical structure: the root node of each hierarchical tree stands for the root identity for a given subject (e.g., individual or collective) while other nodes refer to partial identities for things (e.g., devices, applications, data, resources). Subjects firstly use root identities to create their identity transactions and submit them to the P2P network formed by fog nodes. Then, some fog nodes (namely, miners) start to validate these transactions and pack them into blocks. Once these identity transactions of subjects are mined into blocks and confirmed, relationships between subjects could be built based on their blockchain identities. The blockchain-based social network is thus established with three types of relationships:

- relationships among subjects (S2S), which are similar to relationships in online social networks like Facebook.

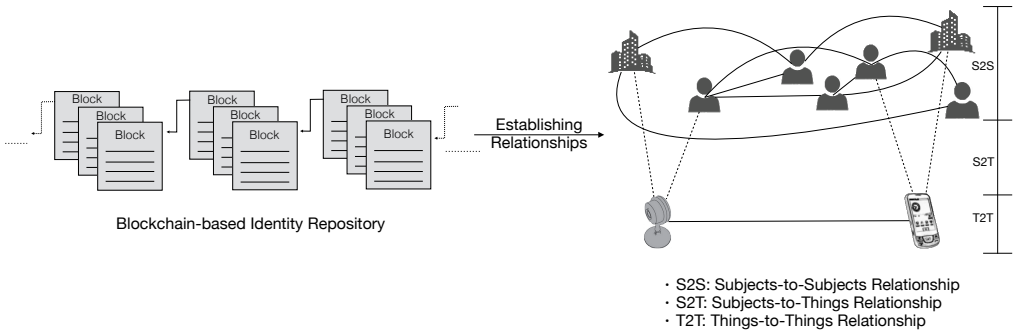


Figure 3.4 – Proposed fog computing paradigm (the relationship management)

- relationships between subjects and IoT things (S2T), which denote ownership of IoT things owned by specific subjects.
- relationships among IoT things (T2T), which designate interactions between services exposed by devices after granting access consent by their subjects.

As shown in Figure 3.4, IoT entities (i.e., individuals, collectives, and things) can be universally unified under the blockchain-based and socialized IoT system model through these relationships, which not only helps to solve the scalability, heterogeneity problems but build user-centric security and privacy solutions. Relationships among subjects make it possible for individuals or collectives to freely build or join social networks and interact with each other. By such, they could form scalable networks at a large scale. Relationships between subjects and IoT things allow owners to set security policies to secure their own devices or services through ownership. Relationships among IoT things are the key to unify heterogeneous things. IoT things could thus provide their services through IoT things relationship management.

Figure 3.5 illustrates our FoCuS from perspectives of the SOA, asynchronous and non-blocking processing architectures (event-driven model), which is composed of many services such as the identity access control management (IAM) service, blockchain-based identity provider service, device management service, threat detection service, and dashboard service. The FoCuS follows the event-driven model and services communicate using asynchronous and non-blocking event bus, which ensures real-time responses, boosts throughput and adaptability to dynamic changes in the context of IoT. In the FoCuS infrastructure, the IAM is the key service, through which subjects could manage their hierarchical identities, create root identity transactions, maintain relationships with other subjects, and configure access control policies through the IAM RESTful interfaces. The blockchain based identity provider (BIdP) driver is responsible for generating blockchain identity transactions according to root identities. After that, the BIdP driver will post these transactions to the BIdP service by calling identity service interfaces in the BIdP. The BIdP service, as the identity provider which can accept bitcoin identity transactions, allows subjects to register, update, revoke and lookup their blockchain identities as verifiable anchors. The device management service offers the access point to IoT devices and is responsible for managing data generated by IoT devices while the things management in the IAM service refers to 1) managing not only devices (i.e., physical objects) but also services

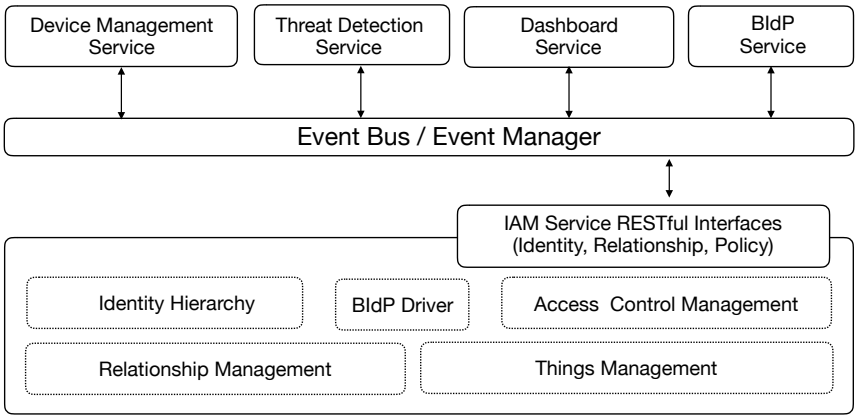


Figure 3.5 – Event-driven FoCuS with REST APIs

(i.e., virtual objects), and 2) only involving identity-related information during the lifecycle (e.g., register, usage, update, and delete) of things. Relying on the IAM service, subjects manage their IoT devices, monitor all services through a dashboard, and detect intrusions by analyzing devices data. In order to describe the structural functionalities of the proposed FoCuS, we illustrate the FoCuS infrastructure from the following two frameworks:

- Blockchain-based identity management framework is in charge of identity management including the hierarchical identity management, blockchain based identity provider, and identity authentication protocol.
- Decentralized IoT access control framework, reveals how identities and relationships are used to design access controls and how threat detection ensures legitimate accesses through blocking threats at runtime.

3.3.1 Blockchain-based Identity Management Framework

As the fundamental component in the FoCuS infrastructure, the blockchain-based identity management framework takes advantage of the blockchain to build a trustless identity provider in distributed P2P networks. Roughly speaking, the blockchain-based identity management framework provides models, algorithms, and protocols to manage hierarchical identities for each subject. Subjects firstly generate their identities through the 3-tuple identity generation template:

$$IDentity =< identifier, credentials, attributes > \tag{3.1}$$

Following the Bitcoin Improvement Protocol (BIP32), we make use of a seed to render identity information (identifiers and credentials). Besides, we adopt reliable double key pairs: one is offline key pair (sk_f, pk_f) stored in the offline devices to prevent from being compromised; another is online key pair (sk_o, pk_o) which is detached from offline key tree branch, randomly generated, could be bound to each offline key pair and used for online operations. The identifier equals the hash value of the offline public key using the double hash function which is a combination of *RIMPE160* and *SHA256*, and corresponding attributes can be attached to each identity. By such, subjects can create their hierarchical identity structure as shown in Figure 3.3.

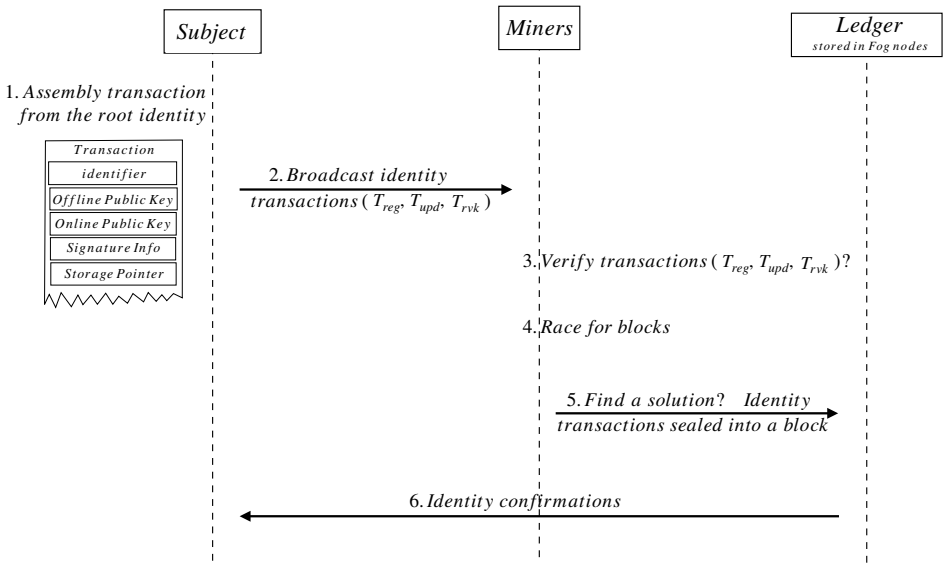


Figure 3.6 – Identity transactions confirmed by the blockchain

After generating hierarchical identities, subjects can use generated root identities to construct blockchain identity transactions and submit these transactions to the peer-to-peer blockchain network. Once corresponding transactions are sealed into blocks and confirmed, relationships could be established based on these identities. The life cycle of subject identities comprises registration, update, revocation, and lookup operations.

- **Identity registration operation.** Subjects who use FoCuS need firstly to register their root identities by sending the identity registration transactions to P2P blockchain networks. After gathering all identity registration information including identifiers, offline public keys, online public keys, signatures, and storage pointers, subjects encapsulate this information into on-chain registration transactions (T_{reg}) and then dispatch the transaction to P2P blockchain networks. Through verification of the registration transaction, fog nodes acting as miners pack identity transaction into a block and broadcast the block to the entire P2P blockchain networks. Once the block is confirmed by the blockchain, the identity registration operation is succeeded. The on-chain transactions verification is depicted in Figure 3.6.
- **Identity update operation** refers to the update of online public key information and storage pointer stored in the blockchain. We use online and offline key pairs to increase the reliability of identity management. The online key pair is used to authenticate identity while the offline key pair is for online key pair updating and the identity revocation. After subjects sending the on-chain identity update transaction (T_{upd}) to P2P blockchain networks, miners will verify the identity update transaction, pack the transaction into a block, and then broadcast the block to the P2P blockchain networks. Once the block is confirmed by the blockchain, the identity update operation is succeeded.
- **Identity revocation operation** means transferring the original identity controlled by an obsolete private key to a new identity, which needs the

signature from online and offline private keys. Once subjects send the on-chain identity revocation transaction (T_{rvk}) to the P2P blockchain networks, miners verify the identity revocation transaction, pack the transaction into a block, and then broadcast the block to the entire P2P network. When the block is confirmed by the blockchain, the identity revocation operation is succeeded.

- **Identity lookup operation** is used to authenticate genuine identities of subjects. The identity lookup is an off-chain operation, which is the foundation of RM and other security services. Subjects access to the blockchain identity ledger which could be stored by any fog nodes, to verify the root identities.

3.3.2 Decentralized IoT Access Control Framework

After having root identities in the blockchain, subjects establish mutual relationships, which directly results in the formation of the socialized IoT network. We propose a relationship management model which comprises three different levels of relationships: Subjects-to-Subjects (S2S), Subjects-to-Things (S2T), and Things-to-Things (T2T).

Suppose $G =< V, R >$ is a social network, and V stands for the set of all social entities while R represents the set of all relationships over social entities. In our relationship management model, we use the social network: $\mathcal{G} =< \mathcal{V}, \mathcal{R} >$ to manage all types of relationships in the IoT environments, where $\mathcal{V}$ only represents the set of subjects and $\mathcal{R}$ refers to the relationship between subjects. Other entities like devices, services could be correlated with their owners (subjects) through ownership (S2T) while T2T could be regarded as relationships between subjects who are behind these things. Due to the introduction of the collective concept, we can use the binary relation to express all direct relations among social entities or their identities. Therefore, we have, given relationship as binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ and $a, b \in \mathcal{S}$, we write $a\mathcal{R}b$, $iff(a, b) \in \mathcal{R}$.

The formation of a relationship between two subjects requires a negotiation procedure to establish an explicit trust relationship in trustless IoT environments as depicted in Figure 3.7. In order to express relationships, we design the relationship declaration structure to include a header (version, identifier, label, timestamp, valid date, data field hash value, signatures) and a payload (capability connectors and data field). In the capability connectors field, we designate inter-operable users so that owners of services could authorize corresponding users to accessors.

When an accessor requests services from IoT devices as shown in Figure 3.7, the relationship needs to be established between the owner and accessor and the owner should make the authorization decision whether or not to grant the privilege. The proposed access control scheme could be summarized from the following steps:

- **Establish a relationship**. Initially, two subjects (the owner and accessor of services) need to negotiate terms to see if the accessor could meet requirements prescribed by the owner. If so, the relationship could be established

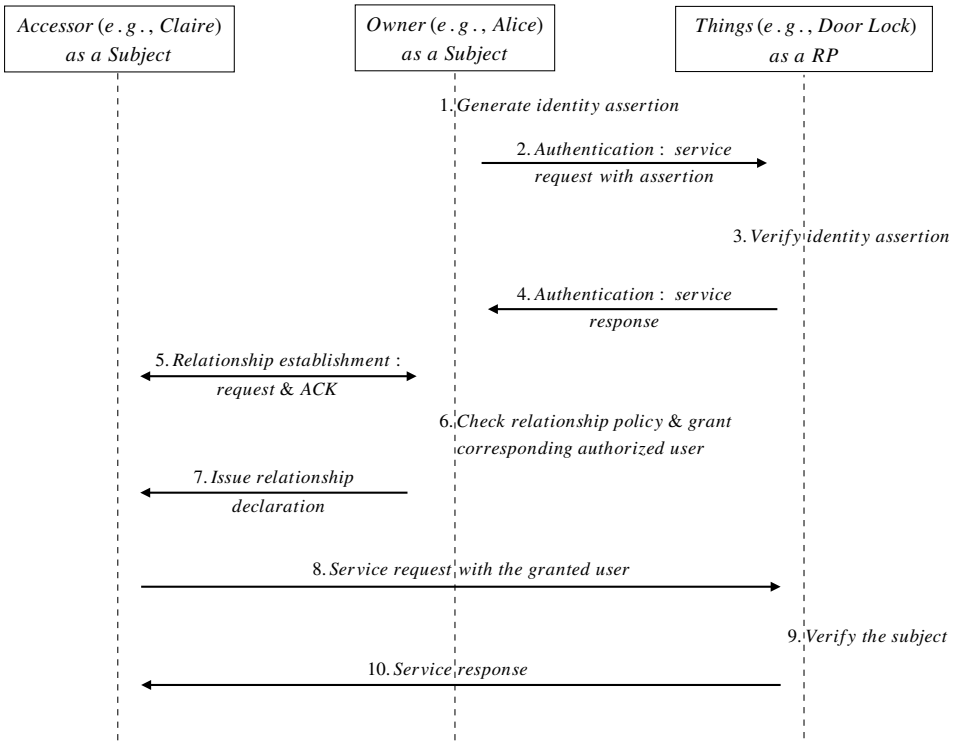


Figure 3.7 – Identity validation between subjects

successfully through adding signatures signed by the root identity online keys of the two subjects.

- **Designate users**. In each established relationship, two partial identities (users) come from two subjects will be mapped as the access representatives of the service in the context of the other.
- **Configure authorization policies**. Based on the established relationship and designated users, the owner could decide what kind of rights or which services should be delegated to the corresponding user through defining roles to the delegated partial identities.
- **Access services**. The accessor could send service requests coupled with the user's information to access services provided by IoT devices.

Besides, we also develop a threat detection service which can continuously monitor anomaly behavior of device operations to protect operations of IoT devices against security threats. It tries to create a reference model for each smart object that describes its normal behavior. The strategy of the autonomic threat detection can be summarized as 1) a training phase to create the reference model, and 2) a testing phase to perform behavior analysis with this model.

3.4 Case Studies

In order to better understand the FoCuS, we give a set of futuristic scenarios which are related to smart home and smart health care domains. These scenarios respectively demonstrate:

- How to manage hierarchical identities as an individual and authenticate an identity under the same subject authority domain (e.g., Smart Home Scenarios).
- How to establish relationships and grant access policies among individuals or collectives.
- How to manage hierarchical identities as a collective and delegate rights between different partial identities under the same collective authority domain. (Hospital Scenario).
- How to establish relationships and grant access policies between an individual and a collective.

3.4.1 Case 1: Hierarchical Identity Tree of Alice

Alice, as an individual, intends to use FoCuS to securely manage all identities and interactions with others in her daily life. Firstly, Alice generates her hierarchical identity tree comprising identities as shown in Figure 3.8, each of which follows the 3-tuple identity generation template in Equation 3.1 to assembly identifier, credentials, and attributes. The root of the hierarchical identity tree is called root identity (RId) of representing the subject individual – Alice while others stand for partial identities (PId) assigned to IoT things according to different application domains (e.g., smart home scenario). From Figure 3.8, we can see, Alice creates partial identities for her car, home, wearables, payment account domains. In each domain, Alice could create the next level partial identities for sub-domains. For instance, there are many partial identities for the main door, garage, home appliances in Alice home scenario. In the car domain, Alice creates sub-domain partial identities for the engine, location, parking, mileage indicator, and remote over-the-air software updates. In wearables domain, Alice creates partial identities for her wearable medical devices such as the cardiac pacemaker and cardiorespiratory activity sensor, which monitor her body parameters (e.g., heart rate, respiration activity) and provide supplementary medical data to her doctor. When trying to access services provided by IoT things, Alice directly uses the corresponding keys to authenticate the corresponding identity as shown in Figure 3.7 Step 1-4. For instance, there is a scenario in Alice authority domains:

Scenario 1: lock or unlock the main door using her cellphone; lock or unlock garage door using her car.

Alice can directly use the main door key and garage door key stored in her cellphone and car respectively to open and close them. After generating the hierarchical identities, Alice still needs to construct blockchain identity transactions using her root identity (RId) and submit them to the P2P blockchain networks composed of fog nodes. Once miners mine the identity transaction into block and Alice gets enough confirmations from the network, her identity is admitted by the peer-to-peer network. Based on the root identity stored in the blockchain, Alice can establish relationships and manage access permissions.

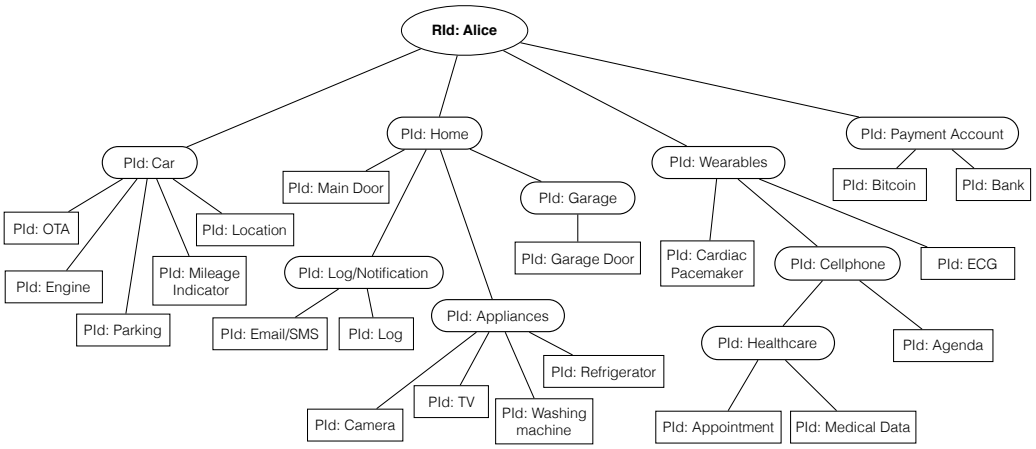


Figure 3.8 – Hierarchical identity tree of Alice

3.4.2 Case 2: Relationships Among Individuals or Collectives

Similarly, other individuals can generate their hierarchical identity trees following the example in Case 1. For instance, Claire creates her hierarchical identity tree, and we give a part of her identity tree (RId: Claire –> PId: Wearables –> PId: Cellphone), which can be used in the following scenario:

Scenario 2: Alice hires a housekeeping – Claire to clean her house regularly and hence Claire needs to open the door of Alice's home.

As shown in Figure 3.7 Step 5-7, Alice and Claire (two individuals) establish a relationship through negotiation, which confirms the relation information and adds signatures signed by the root identity online keys from both parties. At the same time, the corresponding users or partial identities (namely, the partial identity of Alice's house and Claire's cellphone partial identity) are designated. Base on partial identities, Alice grants the corresponding privileges to Claire's partial identity through defining roles comprising accessible services. As for establishing relationships between two collectives, the process is identical with establishing relationships between two individuals.

3.4.3 Case 3: Hierarchical Identity Tree of a Hospital

For individuals, every individual subject can create several identities separately used in different domains. However, for collectives, the collective subject (e.g., hospital), involves more complex domains and sub-domains, in which partial identities can delegate rights to each other. For instance, the hospital collective, as shown in Figure 3.9, has many sub-domains such as hospital departments, patients management, and payment accounts. Doctors, nurses, and medical equipment asset are well organized according to different hospital department centers such as therapeutic center, diagnostic center, and emergency center. For example, the diagnostic center has doctors, nurses, and medical devices like CT Scan, Ultrasound, and Magnetic Resonance Imaging (MRI) Scan. In diagnostic doctors domain, the hospital can create partial identities for each doctor. Therefore, the scenario in the hospital collective case would be:

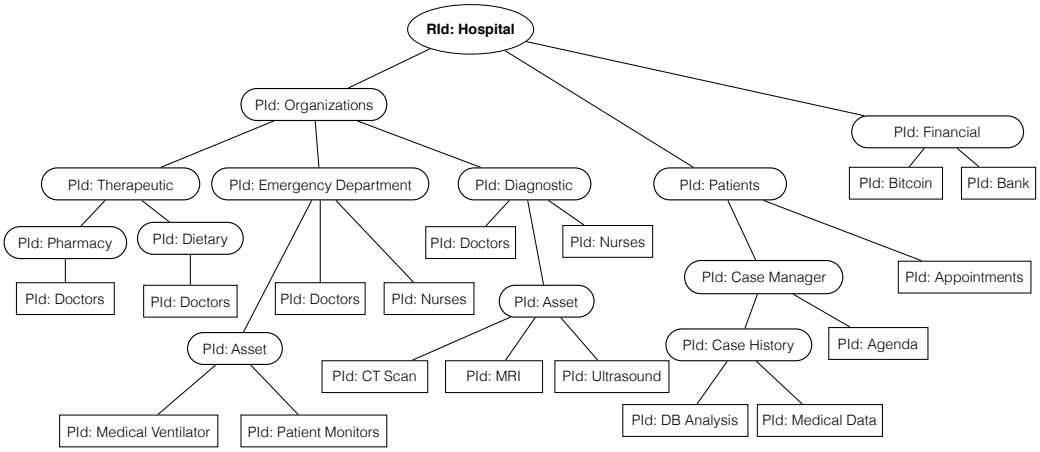


Figure 3.9 – Hierarchical identity tree of a hospital

Scenario 3: The hospital grants a partial identity in doctors domain (namely, a doctor) a set of permissions to access specific medical equipments in asset domain.

When granting access permission between partial identities in one collective hierarchical identity tree, the collective follows the access control frameworks as shown in Figure 3.7 Step 5-7 to establish the things-to-things (T2T) relationship and designate users. The difference is that the owner and accessor are the same subject. The rights delegation process inside a collective can be used in the following individual's scenario:

Scenario 4: Alice generates a partial identity domain in her hierarchical identity tree, which can be used for her young son who has no full capacity for civil conduct.

3.4.4 Case 4: Relationships Between an Individual and a Collective

As mentioned in previous cases, the collective can create partial identities of representing other individuals like doctor identities in the hospital doctors domain. However, created partial identities in the collective still need to be connected with the real individual through establishing relationships between the collective and individual.

Scenario 5: The hospital grants access permissions to individual doctor Bob.

As illustrated in Figure 3.10, individual Bob, as a doctor, establishes the first S2S relationship (e.g., employer-employee) with the hospital and designates corresponding partial identities so that Bob can get the corresponding access permissions in the hospital authority domain.

Scenario 6: Alice takes an appointment in the hospital and creates medical data sharing with the hospital.

As illustrated in Figure 3.10, individual Alice, as a patient, establishes the second S2S relationship (e.g., patient-hospital) with the hospital and designates corresponding partial identities so that Alice can take an appointment and designates partial identities for her medical data with the hospital.

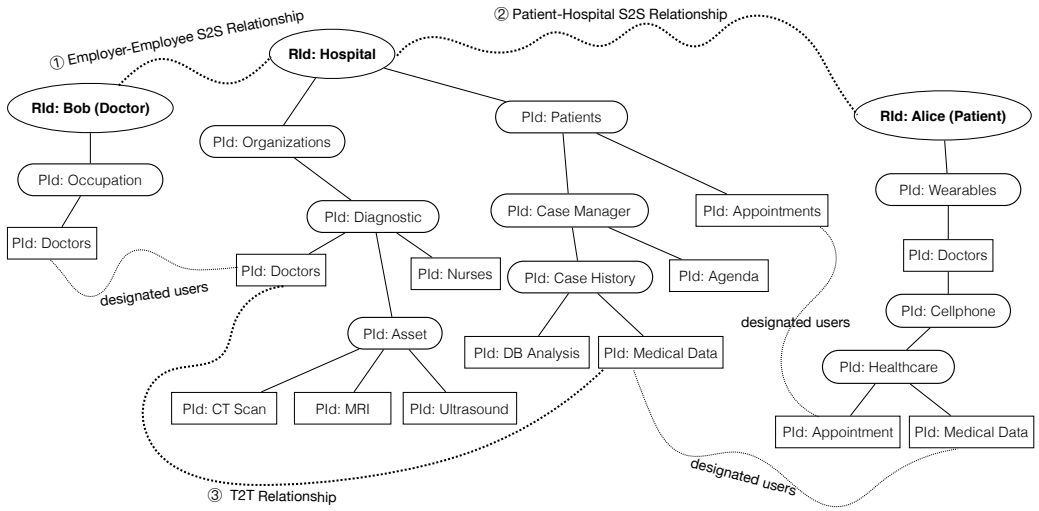


Figure 3.10 – Relationships between individuals and a collective

Scenario 7: The hospital assigns a doctor in the diagnostic center to do the body check for Alice.

As illustrated in Figure 3.10, the hospital establishes the third T2T relationship inside the collective and designates corresponding partial identities so that the doctor can access her medical data in the hospital.

3.5 Summary

In this chapter, we briefly introduced our proposed fog computing secure infrastructure called FoCuS, in which we took advantage of the blockchain, social IoT, SOA, and asynchronous and non-blocking model to build an IoT secure infrastructure with user-centric security and take into account IoT unconventional characteristics. As mentioned before, the fog computing paradigm provides excellent advantages in terms of latency, bandwidth, and mobility. Due to a geographically distributed computing paradigm at the edge of IoT networks, fog computing provides low latency feedback, high bandwidth, and location-awareness services to the vicinity of IoT devices. Therefore, we introduced blockchain-based identity management and decentralized access control frameworks for IoT within the fog computing paradigm. The socialized FoCuS composed of all IoT entities ranging from individuals and collectives to fog nodes, devices, things, and services, makes the IoT extensible and ubiquitous social networks.

Compared with traditional centralized social networks such as Facebook, which compromise user privacy by collecting massive personal data every day and serving as a central trusted identity provider to access Internet services, our proposed blockchain-based identity management framework in FoCuS creates a distributed public ledger used for recording identities and stores them on peers in trustless peer-to-peer networks. All personal data is saved in the distributed network, and only owners have the right to dispose of it. In other words, users retake the control of their information from the so-called “trusted” central companies. Relying on identifiable individuals and collectives (i.e., companies, families, organizations), our FoCuS makes use of social relationships to unify all IoT entities including

devices, applications, data, services. Furthermore, relationships between social entities in the IoT context contribute to building a user-centric decentralized access control management framework where users can use social relationships to set and describe their security policies in trustless IoT environments. Besides, following the SOA principle, we define a set of REST APIs which makes the FoCuS modular and loosely coupled. More importantly, the event-driven FoCuS ensures real-time responses, boosts throughput and adaptability to dynamic changes in the context of IoT. We also described a set of futuristic cases and scenarios which demonstrated how a subject (individual or collective) manage their identities and grant access permissions through establishing relationships with others.

However, FoCuS, as shown in Figure 3.2, is only a proposed SOA-based fog computing security component which ensures security and preserves the privacy of users in IoT environments. It only provides reusable security services, such as identification service, authentication service, access control service and threat detection service for IoT. Therefore, the FoCuS still needs to rely on other SOA-based fog computing modular services such as the virtualization over hardware, devices management, interfaces with cloud servers.

CHAPTER 4

Proposed Blockchain based Identity Management Framework

Contents

4.1 Introduction

Cybersecurity in IoT relies on the digital identity concept to build security mechanisms such as authentication and authorization. As a result, the identity management framework is the priority of building our IoT secure infrastructure. However, current centralized identity management systems are built around third-party identity providers, which raises privacy concerns and presents a single point of failure. Besides, IoT unconventional characteristics such as scalability, heterogeneity, and mobility require new identity management systems to operate in distributed and trustless environments. In order to deal with these challenges, we present our Blockchain-based Identity Management System for the Internet of

Things (BIMSIT) framework. By such, things and people can self-manage their identities and authenticate without relying on any third parties.

The BIMSIT framework mainly relies on the blockchain technology, which is initially introduced by the Bitcoin crypto-currency. By using the blockchain as a back-end technology, we build a distributed and peer-to-peer IoT identity management system in a trustless environment by which all things and people can self-manage their identities and enable authentication without relying on any third parties. Specifically, our BIMSIT framework achieves the following contributions:

- **Hierarchical Identity Information Management**, which adopts the cryptographic paradigm in the bitcoin hierarchical deterministic wallet to redesign the identity data structure, meets the requirements of scalability and interoperability, gives all entities verifiable identities in trustless IoT environments.
- **Distributed Blockchain-based Identity Provider**, which enables the BIMSIT framework with trustless, distributed, and immutable public ledger that records users' digital identities. By such, the BIMSIT overcomes the single point failure and acts as a trusted Identity Provider IdP in trustless IoT networks. Furthermore, the blockchain-based identity provider also simplifies the design of authentication mechanisms via reducing the exchange sequences of "challenges" and "responses" messages.
- **Security and Privacy Analysis**, the BIMSIT framework relies on the threat model, to evaluate cryptographic key compromise, verify the correctness of the simplified authentication protocol using the BAN logic, and analyze security and privacy properties.

The remaining sections are organized as follows. In the preliminaries section, we introduce basic identity concepts and cryptographic building blocks used in the BIMSIT framework. Section 3 introduces the system and threat models for our blockchain-based identity management framework which mainly comprises three parts: algorithms for generating hierarchical identities, blockchain-based identity provider for registration, update, revocation, and look-up of the root identity, and identity recovery from key loss and compromise. Section 4 is fully dedicated to correctness verification analysis, security and privacy analysis of our framework.

4.2 Preliminaries

In this section, we briefly supplement some concepts of the cryptography and Bitcoin transactions used in our blockchain-based identity management system. The related formal notations are described in Table 4.1.

4.2.1 ECDSA Public-Key Cryptosystems

Bitcoin adopts one of the Abstract-Koblitz Elliptic Curves [Kob91] from the Elliptic Curve Digital Signature Algorithm (ECDSA) family called $secp256k1$, pointed out by the Standards for Efficient Cryptography Group (SECG) [Cer99], which is specified by the 6-tuple $T = (p, a, b, G, n, h)$ and the following curve equation:

Table 4.1 – Notations

Notations	Descriptions
$\mathcal{G}$	*identity's identifier generation function*
$\mathcal{R}$	*(pseudo) random number generator*
$\mathcal{H}_A$	*hash function using A algorithm*
sk, pk, σ	*private key, public key, and signature value*
$DROP$	*remove the top stack operant*
DUP	*duplication operation*
$HASH160$	$RIPEMD160(SHA256())$ *function*
$EQUAL(-VERIFY)$	*verify that two parameters are equal*
$CHECKSIG$	*verify a signature using a public key*
$CHECKMULTISIG$	*verifying the given minimum number of signatures against a list of public keys*
$IF-ELSE-ENDIF$	*flow control operators*
$P\mid\equiv X$	P *believes* X
$P \lhd X$	P *sees* X
$P\mid\sim X$	P *said* X
$P\mid\Rightarrow X$	P *controls* X; P *has jurisdiction over* X
$\#(X)$	*the formula is fresh*
$P \overset{K}{\longleftrightarrow} Q; K_{P,Q}$	K *is the shared symmetric key between* P *and* Q
$K \mapsto P$	K *is the public key of* P
$\{X\}_K$	*the cipher text using key* K

$$y^2 = (x^3 + ax + b) \; over \; \mathbb{F}_p \tag{4.1}$$

where the finite field $\mathbb{F}_p$ is defined by $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$ and other parameters in that sextuple are all predetermined constants. Therefore, the $secp256k1$ cryptosystem could be described as follows.

Key Generation. The private key sk, usually generated by a random number generator, is a 256-bit entropy (randomness) while the public key pk is the multiplication between the random private key sk and the predetermined point on the curve called the generator point G.

$$sk = \mathcal{R}(256); \; pk = sk * G \tag{4.2}$$

Signature Generation. Let m be a message to sign, and the signature value σ could be calculated through function $sig(sk, m)$. The signature calculating process works as follows:

- Select a random integer i, $1 \leq i \leq n-1$.
- Compute $iG = (x_1, y_1)$, $r = \bar{x}_1 (mod \; n)$, if $r = 0$, back to the first step. [1]
- Compute i^{-1}, statisfying $i^{-1} * i = 1 \; (mod \; n)$.
- Compute $e = \mathcal{H}_{HASH160}(m)$ and then $\bar{e}$, if $\bar{e} = 0$, back to the first step.

[1] $\bar{x}$ means converting x to an integer.

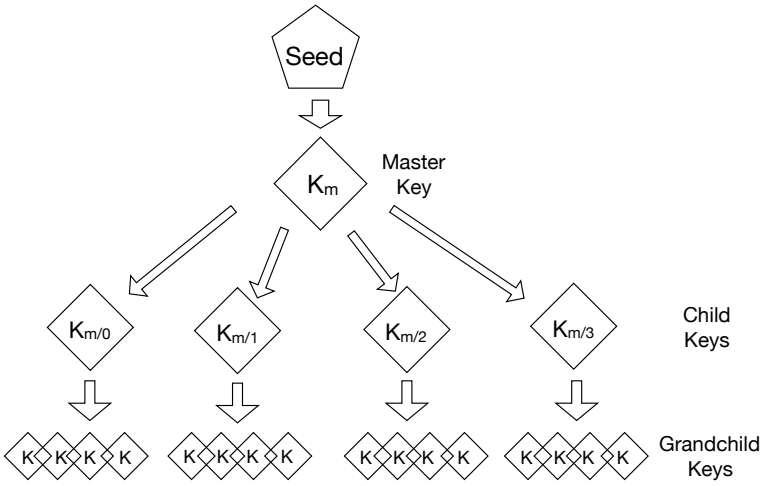


Figure 4.1 – Hierarchical keys in Bitcoin hierarchical deterministic wallet

- $t = k^{-1}(\bar{e} + sk * r)(mod\ n)$.

The signature σ on message m equals (r, t).

Signature Verification. Similarly, we define $verify(\sigma, pk)$ as the function using public key pk to verify the signature on message m. The verification process works as follows:

- Verify the r and t are integers and $1 \leq r, t \leq n - 1$.
- Compute $e = \mathcal{H}_{HASH160}(m)$, $\bar{e}t^{-1}\ (mod\ n)$, $rt^{-1}\ (mod\ n)$, $X = (\bar{e}t^{-1})G + (rt^{-1})pk$.
- If $X = O$, reject. Otherwise, convert the x coordinate of X to an integer $\bar{x}$. If $\bar{x} = r$, accept.

4.2.2 Cryptographic Hash Functions

The cryptographic hash algorithms, involved through the rest of this chapter, are listed as follows. In Bitcoin, two cryptographic hash functions are widely used: SHA256 (Secure Hash Algorithm) [Ant14] and RIPEMD160 (RACE Integrity Primitives Evaluation Message Digest) [DBP96]. For instance, the deriving of Bitcoin addresses (A) from public key pk adopts these two algorithms:

$$A = RIPEMD160(SHA256(pk)) \tag{4.3}$$

In the hierarchical deterministic wallet scheme, the Hash-based Message Authentication Code(HMAC), specifically the HMAC_SHA512 [HE], is used to derive the hierarchical structure of keys. As depicted in Figure 4.1, the hierarchical key structure is generated by a random seed number (128, 256 or 512 bits) through the HMAC_SHA512 algorithm. The parameters of the HMAC_SHA512 algorithm are “key”, which could be the parental key or a string like “Bitcoin Seed”, and “data”, that is, message that you want to calculate.

$$val = HMAC\_SHA512(key, data) \tag{4.4}$$

4.2.3 Blockchain Transactions

As the key concept of the Bitcoin system, transactions are basic units of transferring the bitcoin value. Almost all of the operations in Bitcoin are relevant to transactions so that transactions can be created, broadcast, propagated, validated, and finally added to the transaction ledger [Ant14]. In our blockchain identity management framework, we adopt the transaction structure in the Bitcoin system to encode our modified blockchain-based identity management solution. The structure of a transaction in Bitcoin could be divided into four fields: version, inputs, outputs, and locktime as shown in Figure 2.1. Moreover, inputs and outputs are critical parts to define different types of transactions such as Pay-to-Public-Key-Hash (P2PKH), Pay-to-Public-Key (P2PK) or Pay-to-Script-Hash (P2SH). Therefore, it is possible to add and modify inputs and outputs in any type of transactions like P2PKH in order to later create new types of transactions in our contributions. In Bitcoin, inputs and outputs are written in a set of stack-based forth-like scripts [Ant14], which specify validation operations in transactions and fall into two script categories: the locking script from outputs field called the $scriptPubKey$, and the unlocking script from inputs field called the $scriptSig$. In the BIMSIT framework, we take advantage of the Bitcoin P2SH type transaction to implement our identity management functionalities through defining our identity transactions. All the related notations are illustrated in Table 4.1.

4.3 Blockchain-based Identity Management for IoT

In this section, we first describe our identity system model which plays a significant role in unifying all IoT entities and present the threat model which supports the security and privacy analysis of our framework. Then, we present how the identity hierarchy structure is generated, and how each subject can use the Bitcoin blockchain as the identity provider without relying on any third parties to register and manage digital identities.

4.3.1 Identity System Model

Since the proposed FoCuS infrastructure aims to interconnect all entities in IoT environments securely, the identity system model of the BIMSIT framework should have the capability of assigning identities to all IoT entities. Therefore, we firstly introduce some IoT-related definitions and assumptions in order to describe the identity system model.

Roughly speaking, IoT entities refer to uniquely identifiable everything [ITU09]. In our context, an entity could be individuals (i.e., human beings), collectives (e.g., companies, organizations, families), or things falling under the rubric of the IoT. Therefore, we define:

Definition 1 *(Entity): All identifiable elements like people, organizations, devices, and services in the context of IoT are called entities.*

Assumption 1 *We assume that IoT entities could be classified into two types: Subjects or Things.*

Subjects refer to parties, typically individuals (i.e., human beings) or collectives (e.g., companies, organizations, families), who possess digital identities used for transactions within different domains. Hence, we have:

Definition 2 *(Subject): A subject denotes individuals (i.e., human-beings) or collectives (e.g., companies, governments, organizations) who have a one-to-one correspondence with the juridical person in social life.*

Things denote objects in the physical world (e.g., computers, phones, sensors) or virtual things of the cyber world (e.g., applications, digital resources) [ITU12a]. Therefore, we define:

Definition 3 *(Things): Things refer to entities, which are not subjects. They are divided into Physical Things as the carrier of services or Virtual Objects (a.k.a Virtual Things) such as data, applications or interfaces.*

Given the concepts: *Entity*, *Individuals*, *Collectives*, and *Things* to respectively denote the set of entities, individuals, collectives and physical or virtual things, we have:

$$Entity = \{e | e \in Individuals \vee Collectives \vee Things\} \tag{4.5}$$

The "digital Identity" concept refers to a set of information used for uniquely identifying an entity in a given context. Generally, a digital identity, or an identity for short, comprises three parts [ITU09]: identifier (i.e., UserID, Email, URL, Social security number, ...), credentials (i.e., Certificates, Tokens, Biometrics, Passwords, ...) and attributes (i.e., Roles, Positions, Privileges, Date of birth, ...). Therefore, in our BIMSIT framework, we define:

Definition 4 *(Digital Identity): The digital identity is defined as a 3-tuple structure composed of $identifier$, $credentials$, $attributes$:*

$$IDentity =< identifier, credentials, attributes > \tag{4.6}$$

*in which, the **identifier** is a unique 20-byte hash value (2^{160} possibilities); **credentials** refer to public and private key pairs used for authentication; **attributes** are key-value pair data structure, describing the entity characteristics and its context.*

Based on these definitions, we depict the identity system model from the perspectives of subjects (e.g., people, organizations, companies), domains, root identity, partial identity, stakeholders (i.e., relying parties, identity providers), and relationships among them. Figure 4.2 illustrates these concepts and relationships between different entities. The root identity and partial identities will explained in the following sections.

Subjects are parties, typically individuals (i.e., human beings) or collectives (e.g., companies, organizations, families), who possess digital identities used for

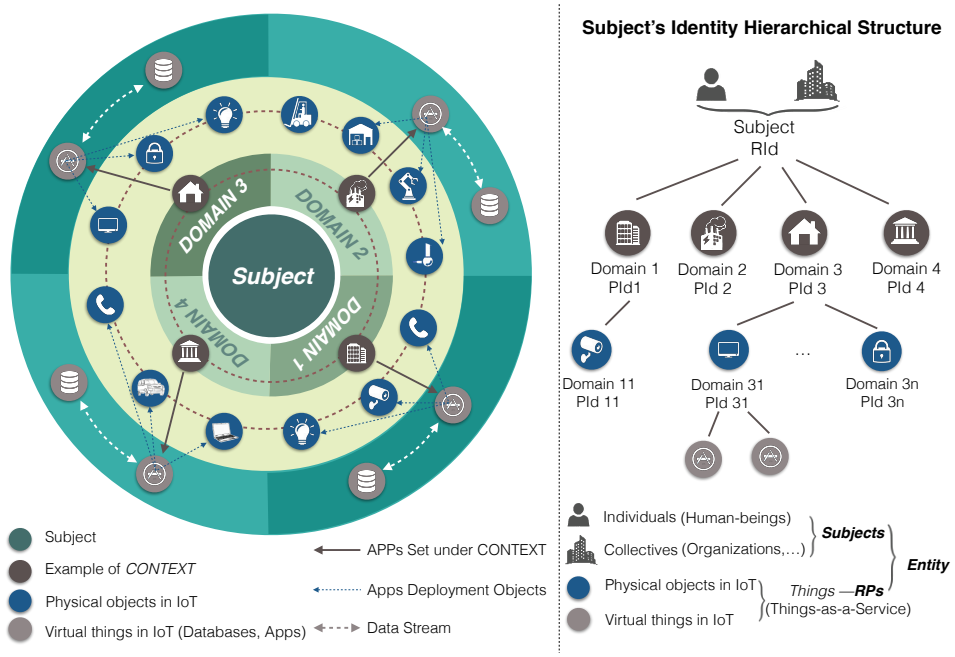


Figure 4.2 – Proposed identity system model

transactions within different domains. Subjects take responsibility for transactions, which could be traced. We use the *Subjects* to denote the set of subjects.

$$Subjects = \{s | s \in Individuals \vee Collectives\} \tag{4.7}$$

Roughly speaking, every individual subject, like employees or inhabitants, could have several digital identities used separately in different security or business domains. Moreover, collective subjects, like a company, may involve departments which possibly have sub-departments as well. For any member of *Subjects*, that is to say, collective subject or individual subject, we introduce the hierarchical structure of identities. The hierarchical identity structure reflects various relations such as organizational, ownership, or security authority relations, among sibling identities or parent-child identities. Besides, we introduce the **Root Identity** (RId) to represent the source identity from which a collective subject or an individual subject generates one or more **Partial Identities** (PId) each of which identifies a domain or sub-domains.

A **domain** is a set of computational and storage resources that are available to a subject, services that are working under the subject authority or, in general, things that are managed by the same subject. Domains can be separated by logical boundaries and architected in a hierarchical manner such as sub-trees in the hierarchical identity structure. The relationship between different domains dictates how two or more subjects can communicate or access resources, services or things of each other. A subject can have several domains each of which is defined as d. The set of all domains is defined as $Domain$. As a result, the entire identity hierarchical structure can be seen as a tree in which each node is expressed by its digital identity and the domain or sub-domain under its control.

The hierarchical identity structure can be produced by the following equations:

$$PId_m = RId \times d_m \tag{4.8}$$

$$PId_{mn} = PId_m \times d_{mn} \tag{4.9}$$

A typical identity hierarchical tree structure is illustrated in Figure 4.2. The subject is defined in terms of its root identity, which can generate partial identities as many as required. Under the root identity, a subject can generate partial identities within specific sub-domains.

Relying Parties refer to parties who are responsible for providing services to *Subjects* and hence called relying parties or service providers. The notation RPs is usually used to denote service providers in identity management systems. In our context, we divide "things" into two categories: physical objects (e.g., computers, sensors) and virtual things (e.g., applications, data, computational and storage capabilities). Physical objects provide services or application programming interfaces (APIs), through which subjects access objects' functionalities and resources. Therefore, we assume, things could be abstracted in term of their services which are the fine-grained levels where access controls and authentications operate:

$$RPs = \{r | r \in Things\} \tag{4.10}$$

In the realm of IoT, accessing and managing things are the fundamental operations. From our perspective, our blockchain-identity management system makes it possible to access and manage things owned by subjects only through services. In this point of view, services are the first-class citizen and could have partial identities to enable fine-grained controls. It is also worth noting that *Things* fall within the ambit of subjects' assets or properties instead of considering them only as subjects as in current Identity Management (IdM) systems, which are extended to cope with IoT. Hence, our identity management system tends to promote user-centric security and covers identity management of subjects and their things as illustrated in Figure 4.2. A Subject is located at the center of the circular representation surrounded by his/her domains defined as smart home, jobs, leisure, and financial accounts. Each domain could designate a partial identity as its authority representative. Therefore, domains are basic logical or structural containers of providing services to subjects. Under a subject's domain, *Things* thus provide services to other subjects through things' interfaces. The ownership between subjects and their things makes it possible to identify *Things*.

Identity Providers are responsible for managing the lifecycle of identities such as identity registration, usage, update, and revocation. In the BIMSIT, the blockchain technology is a core concept upon which we build the Blockchain-based Identity Provider (BIdP) infrastructure, which makes possible to any subject to create and manage their identity in a peer to peer and trustless IoT networks. We use the notation $IdPs$ to denote identity providers in IdM systems.

In summary, IoT identity management systems should be able to manage all identities of entities, such as identities of *Subjects* and *Things*. As depicted in Figure 4.2, the blockchain-based identity management system for IoT (BIMSIT) refers to subjects and their hierarchical identity structure (HIS). Subjects are en-

tities and refer to the individual or collective subjects. The hierarchical identity structure organizes all digital identities that a subject may generate. Each identity defines a domain, comprising all sub-identities that are generated from the corresponding domain. Hence, our BIMSIT could be defined as follows:

$$BIMSIT = \{< Subject, HIS >\} \tag{4.11}$$

$$HIS = \{< Identity, d >\} \tag{4.12}$$

4.3.2 Threat Model

In order to assess the security and privacy properties of our BIMSIT framework, we propose a threat model against which we analyze the security taking into account active adversaries who are trying to impersonate identities of other subjects by replacing genuine public keys with fake ones[FVY14a]. In other words, adversaries could hijack the deployed IoT devices, clone, retrieve, or even modify identity information stored in these devices. Supposing that adversaries are computationally-bounded, they could launch passive attacks such as eavesdropping, reply attack. They also could intercept all communications among parties and altering or blocking messages. Furthermore, we assume, some internal adversaries could retrieve online private keys and impersonate the identity. However, we assume that offline key pairs are securely protected and stored in hardware with Personal Identification Number (PIN) or biometrics. Like hardware wallets in crypto-currencies, the specialized hardware is designed for managing offline keys and signing transactions inside instead of exposing private keys on the Internet directly. The threat model and its security analysis are covered in details in the "Analysis of BIMSIT" section.

4.3.3 Hierarchical Identity Information Generation

In order to generate the hierarchical identity structure (HIS) and its related information (i.e., identifier, credentials, attributes), we take advantage of the cryptographic algorithms used in hierarchical deterministic wallet (BIP32[2]), which provides users with a convenient and straightforward way to backup and recover all keys. Following the BIP43[3], we further define two levels in BIP32 tree structure for interoperable operations with other cryptographic products such as hardware wallets:

$$m/purpose'/accounts'/$$

Under the master key (m), the first level – purpose, is set to constant $200'$ (i.e., 0x800000C8). Therefore, we have 2^{28} available accounts (key pairs) for each subject. As shown in Fig. 4.3, a given subject who wants to rely on our BIMSIT framework to manage digital identities, needs to generate a random number called *seed* which has 128, 256 or 512 bits, and can be encoded into mnemonic words for easy backup. From this *seed*, the subject can obtain the **Root Identity** information (i.e., the identifier and offline key pair). From the root identity offline

[2]https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki
[3]https://github.com/bitcoin/bips/blob/master/bip-0043.mediawiki

key, the subject can create multiple **Partial Identities**, each of which could be used to generate sub-partial identities. The partial identities generation can be repeated recursively at any depth. As depicted in Fig. 4.2, the hierarchical identity of subjects is a tree, including up to 2^{28} (partial identities) each of which is defined in terms of its identifier, credentials, attributes. When a subject to add a new (sub or partial) identity and thus creates a subbranch as a domain (e.g., the partial identity11 in Fig. 4.2), the index number of the new sub or partial identity will be deduced according to the position of the subbranch and used for generating the corresponding offline key pair of the identity. In order to calculate the index number, the identity tree structure should be converted into a binary tree where nodes are expressed as binary numbers. Then, the index number of new identity node is the binary number in the binary tree.

In our identity information generation scheme, we set the value of the identity identifier as the result obtained from a 160 bits double-hash function applied to the offline public key:

$$identifier = HASH160(pk_f) \tag{4.13}$$

In order to enhance the reliability of each identity in the hierarchy, we consider the double key pairs (i.e., online and offline key pairs) as the identity credentials. The offline key pair (sk_f, pk_f) is stored in offline devices to prevent the identity from being compromised while the online key pair (sk_o, pk_o) is detached from the offline key tree branch, randomly generated, and bound to each offline key pair for online operations. We also add the signature segment to prevent the online key pair from being replaced by malicious attackers. The message (m) parameter of the $sig(sk, m)$ is the binary OR operation result between the online public key and self-examination attributes which, we assume, are the return value of hardware interfaces and directly bound to specific hardware and cannot be modified. These attributes are conducive to prevent identity cloning attack of devices. The private key (sk) parameter of the $sig(sk, m)$ is the offline private key of the identity. Therefore, the signature, as a part of credentials, is derived through the following equation.

$$\sigma_{sk_f} = sig(sk_f, pk_o||self\_ex\_attr) \tag{4.14}$$

The subject could also assign additional and valuable information to the identity by creating a list of attributes when the corresponding identity is established. Attributes are particularly useful to set contextual information and constraints to be used by the access control policies and rules. The access controls are explained in the next chapter.

Besides, the identity information generation is responsible for generating **Root Identity** information, which is the stepping stone to construct the system for IoT, and **Partial Identity** information, which could also represent the identity of a thing as the ending of hierarchical identity structure. Therefore, we develop the following **hierarchical identity generation algorithm** to build digital identities for a given subject as follows.

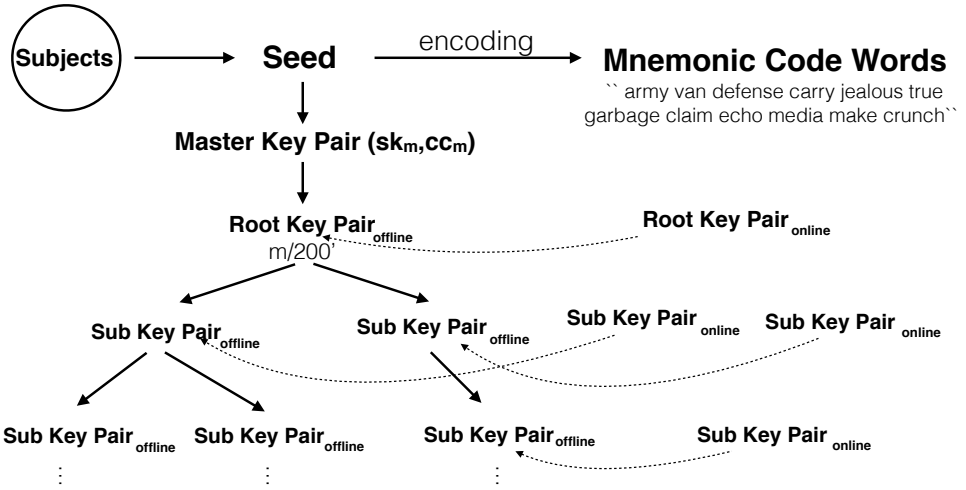


Figure 4.3 – Hierarchical keys in identity information generation

---

Algorithm Generating Hierarchical Identity

participants: Subject s
input: 1) Null, or 2) Root identity with a generating index (binary tree path)
output: Hierarchical identity structure

procedure GEN_HID(s, $identity$, $index$)
  var val, v_L, v_R //Temporary variants
  function $CheckFormat()$ //Check parental identity
  s executes: // s: Subject who wants to generate identity
1: **switch** ($identity.identifier$&$identity.credentials$)
2: **case** NULL**:**
3: $MasterExtKeys = DKD(''HID\_Seed'')$
4: $RKP_{m/200'} = CKD(MasterExtKeys, m/200')$
5: $RandomExtKeys = DKD(''HID\_Random'')$
6: $pk_f = RKP_{m/200'}.pk; sk_f = RKP_{m/200'}.sk$
7: $pk_o = RandomExtKeys.pk$
8: $sk_o = RandomExtKeys.sk$
9: $identifier \leftarrow HASH160(pk_f)$
10: $attributes \leftarrow setAttributevalues(Names, Values)$
11: $\sigma_{sk_f} = sig(sk_f, pk_o||self\_ex\_attr)$
12: $credentials \leftarrow < RKP_{m/200'}, RandomExtKeys, \sigma_{sk_f} >$
13: **case** NOT NULL**:**
14: **if** $(CheckFormat(identity))$ **then**
15: $ek = identity.credentials.RKP_{m/200'}$
16: $SKP_{index} = CKD(ek, index)$
17: $RandomExtKeys = DKD(''HID\_Random'')$
18: $pk_f = SKP_{index}.pk$
19: $sk_f = SKP_{index}.sk$
20: $pk_o = RandomExtKeys.pk$
21: $sk_o = RandomExtKeys.sk$

22: $identifier \leftarrow HASH160(pk_f)$
23: $attributes \leftarrow setAttributevalues(Names, Values)$
24: $\sigma_{sk_f} = sig(sk_f, pk_o||self\_ex\_attr)$
25: $credentials \leftarrow < RKP_{m/200'}, RandomExtKeys, \sigma_{sk_f} >$
26: **end if**
27: **end switch**
28: **return** *identity*

sub-procedure DKD(string)
function $\mathcal{R}$ //(Pseudo) Random number generator
var val, v_L, v_R //Temporary variants
29: $seed = \mathcal{R}(128/256/512)$
30: $val = HMAC\_SHA512$(string, $seed$)
31: $v_L = val[0-255]$; $v_R = val[256-512]$
32: $sk_m = v_L$; $cc_m = v_R$; $pk_m = sk * G$
33: $extended\_keys =< pk_m, sk_m, cc_m >$
34: **return** *extended_keys*

sub-procedure CKD($root\_extended\_keys$, $index$)
const n // one constant parameter of the elliptic curve.
var val, v_L, v_R //Temporary variants
function ser_{32}(i) //serialize a 32-bit unsigned integer i as a 4-byte sequence, most significant byte first.
35: $ek = root\_extended\_keys$
36: $val = HMAC\_SHA521(ek.cc, ek.sk||ser_{32}(index))$
37: $v_L = val[0-255]$; $v_R = val[256-512]$
38: $sk = (ek.sk + v_L) mod\ n$; $cc = v_R$; $pk = sk * G$
39: $extended\_keys =< pk, sk, cc >$
40: **return** *extended_keys*

To sum up, the hierarchical identity generation algorithm is responsible for the generation of digital identities, ranging from root identity to a hierarchy of partial identities including online and offline keys. In formal notation, we define $IDS(s)$ as the **Identity Set** of a given subject s. It consists of $IDS_f(s)$ and $IDS_o(s)$ to respectively denote the set of all offline private keys (sk_f) of s and the rest part of $IDS(s)$ except the $IDS_f(s)$.

$$IDS(s) =< IDS_f(s), IDS_o(s) > \tag{4.15}$$

It is worth noting that the offline key pair has a particular role, as to revoke its corresponding online key pair in case the online private key is compromised. As for the revocation or deleting operation of offline keys, we can add a boolean flag to denote whether the key (or the associated partial identity) is enabled or disabled.

4.3.4 Blockchain-based Identity Provider

In the BIMSIT framework, the essence of identifying *Things* or *Relying Parties* is the identification to subject identities. *Things* as assets or properties are identified through their subject identities. Therefore, the identity management for subjects is the kernel of the entities' identity management in IoT. In this section, we use the Bitcoin blockchain technology to build a trusted public identity provider for all subjects in completely distributed networks without any centralized or decentralized servers. The blockchain-based identity provider is mainly responsible for the following operations: registration, update, revocation, and lookup of subject digital identities. In our contribution, we use the Bitcoin blockchain transaction structure to encode these identity management operations, and hence miners can mine these identity transactions into the blockchain. We mainly update the transaction fields with scripts that implement the registration, update, and revocation operations.

4.3.4.1 Identity Registration

The identity registration operation should be elaborated from three different parties: *subject*, *miners* and *peers*. Firstly, the subject who wants to have a digital identity, needs to organize the identity information generated through the hierarchical identity scheme as described in the previous subsection, which includes root identity's $identifier$, pk_f, pk_o, storage *pointer* of the online identity set and the *signature* of the entire transaction using sk_o and sk_f. The generation of the signature makes use of the default option of generating transaction signatures (i.e., $SIGHASH\_ALL$) in Bitcoin stack, which signs all transaction outputs and inputs without signature scripts. Transaction signatures in update and revocation phases follow the same generation procedure.

By using the identity registration information, the subject completes the identity registration process, in which we make use of the P2SH transaction to firstly create a pre_TX_{reg} transaction in order to include the identity information in the redeem script of TX_{reg}. The unlocking scripts in the pre_TX_{reg} and TX_{reg} denote that the sender has the offline and online private keys while the signature of unlocking and locking scripts means that others could not be able to modify the identity information like the pointer and keys. After assembling the pre_TX_{reg} and TX_{reg}, the subject will post the pre_TX_{reg} and await the transaction mined into a block and then post TX_{reg} to the entire network. Whenever miners or peers validate new transactions, they will verify if the $identifier$ matches the pk_f and the signatures are correct which ensures no data tampering from malicious parties in pk_o and storage *pointer*. In addition to verifying the received TX_{reg}, miners will pack it to the candidate block, do the Proof-of-Work, race for the valid block, and broadcast the block to the entire P2P network. Once the corresponding block is confirmed by enough subsequent blocks, the identity registration succeeds. The following identity registration phase of Protocol BIdP elaborates specific operations from the perspective of subjects, miners, and peers respectively.

Protocol Blockchain-based IdP: Registration Phase

participants: A subject s, Miners, Peers
input: root identity
output: Mined identity registration transactions

procedure ID_REG(s, $root\_identity$)
var $identifier$ //subject's root identity identifier
var pk //subject's root public key credentials
var σ_{sk} //signature of locking and unlocking scripts
var tx_script_info //locking and unlocking scripts

Subject s organizes:
1: $identifier = root\_identity.identifier$
2: $pk_f = root\_identity.credentials.pk_f$
3: $pk_o = root\_identity.credentials.pk_o$
4: $pointer = \mathcal{H}_{HASH160}(IDS_o(s))$
5: $\sigma_{sk} = sig(sk, tx\_script\_info)$
6: Uses $< identifier, pk_f, pk_o, \sigma_{sk}, pointer >$ to construct the TX_{reg}

P2SH pre_TX_{reg}
Unlocking Input Script: $< \sigma_{sk_o} > < pk_o >$
Locking Output Script: $HASH160$
$< TX_{reg} - redeemScriptHash > EQUAL$

P2SH TX_{reg}
TX_{reg} Redeem Script:
$< pointer > DROP\ DUP\ HASH160$
$< identifier > EQUALVERIFY$
$CHECKSIG$
Unlocking Input Script:
$< \sigma_{sk_f} > < pk_f >$
$< TX_{reg} \quad redeemScript >$
Locking Output Script: $HASH160$
$< TX_{upd/rvc} - redeemScriptHash > EQUAL$

7: Posts the P2SH transaction pre_TX_{reg} and awaits the transaction mined into a block, and then posts TX_{reg} to the entire network

Miners execute **Identity Reg Transaction Verification**:
8: check pk_f and $identifier$ satisfying:
$identifier = HASH160(pk_f)$
9: check $tx\_script\_info = verify(\sigma_{sk}, pk)$
10: ***Miners*** race for **Identity Block Formation**:

Peers execute **Verification of Identity** stored in Blockchain Transaction

11: check pk_f and $identifier$ satisfying:
$identifier = HASH160(pk_f)$
12: check $tx\_script\_info = verify(\sigma_{sk}, pk)$

4.3.4.2 Identity Update

The identity update refers to replacing the online public key pk_o or storage *pointer* stored in the blockchain identity transactions. Similarly, the identity update transaction, called TX_{upd}, is generated based on the subject's need to update some information regarding their identities. According to different modification needs, we have identified different authentication levels. If the subject requires to modify the storage *pointer*, the signature using online private key sk_o is needed whereas if the subject requires to replace the online public key, the double signatures using online and offline private keys are thus indispensable. The structure of the TX_{upd} transaction and its operation are performed by the following identity update phase. After subjects posting the identity update transaction TX_{upd} to networks, miners and peers will validate and mine the transaction TX_{upd} into a block. One thing to be noted is, the storage pointer update operation is designed for online backup and hence not mandatory. Users can schedule online backups periodically in that the primary backups can be locally scattered among user's more than one devices.

Protocol Blockchain-based IdP: Update Phase

participants: A subject s, Miners, Peers
input: root identity, modified data (pointer or online key)
output: Mined identity update transactions

procedure ID_UPD(s, $root\_identity$, $modifiedData$)
var new_pk_o // the new online public key
var new_sk_o // the new online private key
var $new\_pointer$ // the new pointer
var σ_{sk} // signature of locking and unlocking scripts
var $modifiedDataType$ // the modified data type

Subject s organizes the modified data:
1: $pk_f = root\_identity.credentials.pk_f$
2: $pk_o = root\_identity.credentials.pk_o$
3: $\sigma_{sk} = sig(sk, tx\_script\_info)$
4: Uses $< pk_f, pk_o, \sigma_{sk} >$ to construct the transaction TX_{upd}

P2SH TX_{upd} Redeem Script:
IF
$DROP < pk_o > CHECKSIG$
$ELSE$
$2 < pk_o > < pk_f > 2\ CHECKMULTISIG$
ENDIF

Locking Output Script: $HASH160$
$< TX_{upd/rvc} - redeemScriptHash > EQUAL$

5: **switch** $(modifiedDataType)$
6: **case** $isPOINTER$**:**
7: $new\_pointer = modifiedData$
8: Unlocking Input Script:
$< \sigma_{sk_o} > < new\_pointer > 1$
$< TX_{upd} \quad redeemScript >$
9: **case** $isPUBKEY$**:**
10: $new\_sk_o = modifiedData$
11: Unlocking Input Script:
$< new\_\sigma_{sk_o} > < new\_pk_o > CHECKSIG$
$0 < \sigma_{sk_o} > < \sigma_{sk_f} > 0$
$< TX_{upd} \quad redeemScript >$
12: **end switch**
13: Subject posts the transaction TX_{upd} to the entire network
14: ***Miners*** execute **Identity Upd Transaction Verification**
15: ***Miners*** race for **Identity Block Formation**:
16: ***Peers*** execute **Verification of Identity** stored in Blockchain Transaction

4.3.4.3 Identity Revocation

The identity revocation means revoking the offline public key in the blockchain, which is achieved by transferring the original identity controlled by an obsolete keypair to a new identity controlled by a new key pair. This operation requires double signatures from the online and offline private keys. After subjects posting the identity revocation transaction TX_{rvc} to networks, miners and peers will validate and mine the transaction TX_{rvc} into a block. The structure of TX_{rvc} could be found in the following identity revocation phase of Protocol BIdP.

Protocol Blockchain-based IdP: Revocation Phase
participants: A subject s, Miners, Peers
input: root identity, offline key
output: Mined identity revocation transactions

procedure ID_RVC(s, $root\_identity$, new_sk_f)
var new_pk_f // the new offline public key
var new_sk_f // the new offline private key
var σ_{sk} //signature of locking and unlocking scripts

Subject s organizes the modified data:
1: $pk_f = root\_identity.credentials.pk_f$
2: $pk_o = root\_identity.credentials.pk_o$
3: $\sigma_{sk} = sig(sk, tx\_script\_info)$

4: Uses $< pk_f, pk_o, \sigma_{sk} >$ to construct the transaction TX_{rvc}

P2SH TX_{rvc} Redeem Script:
IF
$DROP < pk_o > CHECKSIG$
$ELSE$
$2 < pk_o > < pk_f > 2\ CHECKMULTISIG$
ENDIF
Locking Output Script: $HASH160$
$< TX_{upd/rvc} - redeemScriptHash > EQUAL$

Unlocking Input Script:
$< new\_\sigma_{sk_f} > < new\_pk_f > CHECKSIG$
$0 < \sigma_{sk_o} > < \sigma_{sk_f} > 0$
$< TX_{rvc} \quad redeemScript >$

5: Subject posts the transaction TX_{rvc} to the entire network
6: ***Miners*** execute **Identity Rvc Transaction Verification**
7: ***Miners*** race for **Identity Block Formation**:
8: ***Peers*** execute **Verification of Identity** stored in Blockchain Transaction

---

4.3.4.4 Identity Lookup

The identity lookup operation is encapsulated in the off-blockchain transaction TX_{lkp} while the three previous transactions, namely, TX_{reg}, TX_{upd}, and TX_{rvc} belong to on-chain transactions. The difference is that TX_{lkp} does not need to be delivered to miners or peers to execute the verification and is similar to the operation of using the address to locate the $UTXO$ in Bitcoin system. The following lookup phase of Protocol BIdP describes the operation of identity lookup.

Protocol Blockchain-based IdP: Lookup Phase

participants: A subject s, Peers
input: identifier, signature, tx_id
output: True or False (Valid Identity)

procedure ID_LKP(s, $identifier$, σ_{sk_o}, tx_id)
var $blockchainLedger$ //blockchain identity repository
var σ_{sk_o} //signature during authentication process
function $retrievePK(ledger, i, tx\_id)$ //retrieves the public key according to the $identifier\ i$ in blockchain identity ledger

Subject s searches blockchain ledger to get public key
1: $pk_o = retrievePK(blockchainLedger, identifier, tx\_id)$
2: **if** $verify(\sigma_{sk_o}, pk_o)$ **then**
3: Return TRUE
4: **else**
5: Return FALSE

6: **end if**

The identity lookup is a basic operation of other services like authentication and authorization which can be used to authenticate the true identity of subjects and delegate authorization according to different identities. For instance, when subject s_1 gives the s_1's identifier and signature to a relying party whose owner, we assume, is subject s_2, s_2 can verify the s_1' identity through the lookup operation, in which s_1 and s_2 also could be the same subject. The signature is a part of the identity assertion (IA), which will be explained in the following session protocol.

4.3.5 Identity Recovery from Key Loss and Compromise

In the BIMSIT framework, we have four types of secret information: mnemonic code words, hierarchical tree structure, offline private keys, and online private keys. We list all countermeasures in Table 4.2 when the key is lost or compromised.

If the online private keys are compromised, users only need to update the original key with a new one. If the offline private keys are lost, we could recover them using the mnemonic words. Since we have the assumption that it is not

Table 4.2 – Identity recovery from key loss and compromise

	Loss	**Compromise**
Mnemonic Words	–	–
Hierarchical Tree Structure	∗	*Mnemonic Protected*
Offline Private Keys	*Recovery*	–
Online Private Keys	∗	*Update*

possible to compromise offline keypairs through calling APIs and hijacking offline hardware, offline private keys are safe. For the hierarchical identity tree structure and online private keys, we may have many offline and online encrypted copies distributed in many devices; the loss situation is not thus quite possible. Besides, if hackers could get the tree structure of the identity, they still could not be able to retrieve secret identity information in that they do not have mnemonic words (master keys). Like all the BIP32 solutions, the mnemonic code words are too important to be lost or compromised. However, we do have other alternative solutions [Upo] to solve this problem through secret sharing or multi-party computation algorithms.

4.4 Analysis of BIMSIT

According to the previous section, the BIMSIT could be divided into two parts: the hierarchical identity information management and authentication and authorization services. The former comprises the identity generation, registration, update, and revocation while the latter refers to security services such as the authentication which relies on the identity lookup operation as shown in Figure 4.4. Although having stakeholders such as subjects, relying parties, and identity providers, as the same as most of traditional identity management systems, the

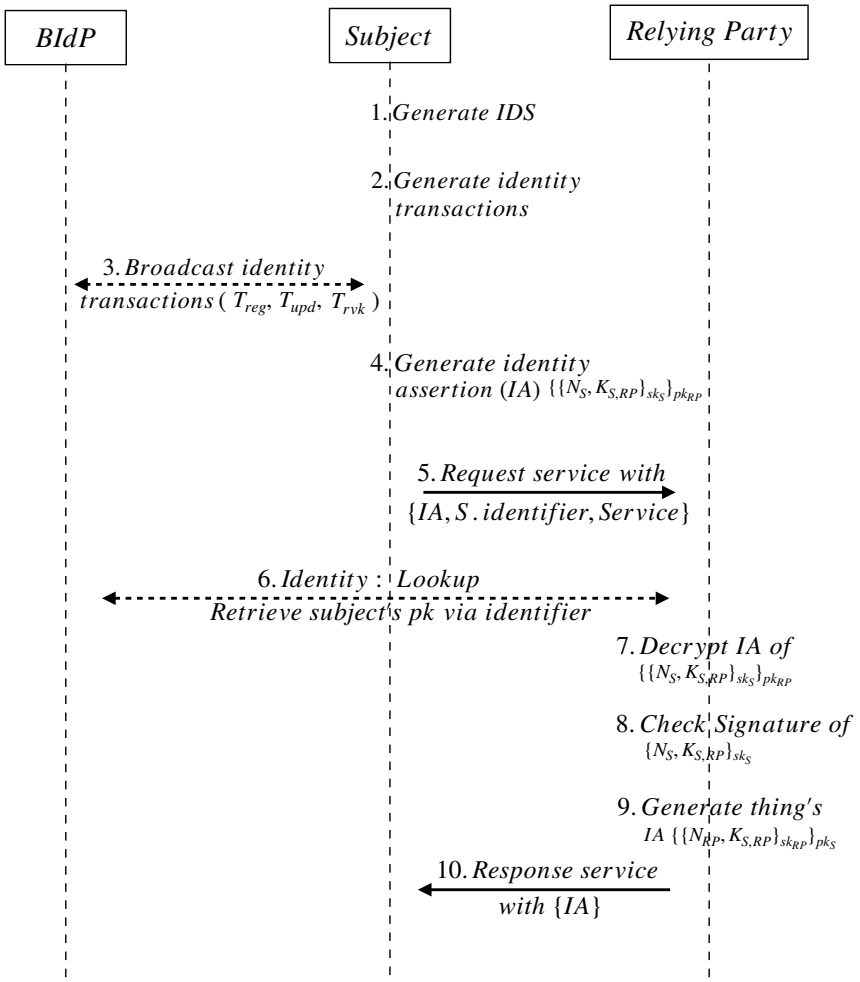


Figure 4.4 – BIMSIT overview

BIMSIT still makes a significant difference in the methodology of managing identities. Because it removes the assumption of users giving full trust to their IdPs and takes back control over their own identity from the so-called "trusted third parties". In the rest of this section, we first verify the correctness of our BIMSIT using the BAN logic [BAN89], which is a set of rules for formally describing and analyzing security protocols. Then, we conduct security analysis over the BIMSIT framework.

4.4.1 Correctness Verification Analysis

All operations and interactions between IdMS stakeholders could be found in Figure 4.4. Compared with traditional identity management systems, our BIMSIT eliminates "trusted" third parties, which makes many operations (i.e., generating session keys) are done by subjects themselves rather than relying on the so-called "trusted" third parties. When two subjects decide to establish a session to allow one subject to access a service provided by another subject (called the relying party), the communication session is established by the subject who initiates the request. In addition, the relying party (RPs), namely, $things$ could get the identity information about the initiator of the service request from their owners

(that is the subject who owns the *things*); therefore, the identity lookup operation (i.e., step 6) in Figure 4.4, could be regarded as internal communication. After generating the identity set IDS, the subject will deliver the previously defined identity transactions to blockchain networks in order to establish the consensus of identity. Once the blockchain network achieves the identity consensus, the subject is allowed to generate the identity assertion (IA), which is composed of identity information and the signature of this assertion so that it could be used to verify the subject's identity. We use the formal BAN logic language to describe the session key establishment protocol in Table 4.3. According to the description

Table 4.3 – Formal description of BISMIT session protocol

Step 1:	$S->BIdP:T_{reg}/T_{upd}/T_{rvk}$
Step 2:	$BIdP->S:ACK_{confirmation}$
Step 3:	$S->RP:N_S,K_{S,RP},\{N_S,K_{S,RP}\}_{sk_S},\{\{N_S,K_{S,RP}\}_{sk_S}\}_{pk_{RP}}$
Step 4:	$RP->BIdP:S.identifier$
Step 5:	$BIdP->RP:S.identifier,pk_{S.identifier}$
Step 6:	$RP->S:N_{RP},\{\{N_{RP},K_{S,RP}\}_{sk_{RP}}\}_{pk_S}$
Session Key:	$K_{S,RP}$

of our BIMSIT session key establishment protocol, we firstly list assumptions of the protocol, delete the unencrypted messages, idealize the only two left messages Message 1 and Message 2 and finally idealize the protocol goals. Please note that the operators are summarized in Table 4.1.

From the description of idealization, we can see, there are only two messages, which includes the encrypted information, needed to be idealized, and the idealized goals are also simplified, which eliminates the goal:

$$S| \equiv S \overset{K}{\longleftrightarrow} RP \tag{4.16}$$

in traditional session key establishment protocols security analysis, since the session key is generated by the initiator of the request. The detailed verification proofs are shown in Table 4.4. Though the verification process, we can deduce these three goals, which ensure that the secured session key is established successfully.

4.4.2 Security and Privacy Analysis

Although the BAN logic could formally verify the correctness of our protocol, it has limitation in terms of security analysis since it cannot analyze security attacks comprehensively. Therefore, we identify a list of possible attacks and threats related to the identity information generation and communication processes in order to apply the security analysis. Besides the identity information management, in the BIMSIT framework, there are three types of communication processes:

- Subjects and Distributed Hash Table (DHT) peers communication
- Subjects/RPs and blockchain IdP communication

Table 4.4 – Formal verification

Assumptions			
A1:	$S, RP\| \equiv pk_S \mapsto S, pk_{RP} \mapsto RP$		
A2:	$S\| \equiv \#(N_{RP}), RP\| \equiv \#(N_S)$		
A3:	$RP\| \equiv S \Rightarrow K_{S,RP}$		
Idealized Message Representation			
M1:	$S- > RP : \{\{N_S, K_{S,RP}\}_{sk_S}\}_{pk_{RP}}$		
M2:	$RP- > S : \{\{N_{RP}, K_{S,RP}\}_{sk_{RP}}\}_{pk_S}$		
Goals			
G1:	$S\| \equiv RP\| \equiv S \overset{K_{S,RP}}{\longleftrightarrow} RP$		
G2:	$RP\| \equiv S \overset{K_{S,RP}}{\longleftrightarrow} RP$		
G3:	$RP\| \equiv S\| \equiv S \overset{K_{S,RP}}{\longleftrightarrow} RP$		
Verification Process			
(1):	$RP \triangleleft \{\{N_S, K_{S,RP}\}_{sk_S}\}_{pk_{RP}}$		$M1$
(2):	$RP\| \equiv pk_{RP} \mapsto RP$		$A1$
(3):	(1), (2) $RP \triangleleft \{N_S, K_{S,RP}\}_{sk_S}$		$Seeing \quad Rule$
(4):	$RP \triangleleft \{N_S, K_{S,RP}\}_{sk_S}$		MP
(5):	$RP\| \equiv pk_S \mapsto S$		$A1$
(6):	(4), (5) $RP\| \equiv S\| \sim N_S, K_{S,RP}$		$Message - Meaning \quad Rule$
(7):	$RP\| \equiv S\| \sim N_S, K_{S,RP}$		MP
(8):	$RP\| \equiv \#(N_S)$		$A2$
(9):	$RP\| \equiv \#(N_S, K_{S,RP})$		$Freshness \quad Rule$
(10):	(7), (9) $RP\| \equiv S\| \equiv (N_S, K_{S,RP})$		$Nonce - Verification \quad Rule$
(11):	$RP\| \equiv S\| \equiv (N_S, K_{S,RP})$		MP
(12):	$RP\| \equiv S\| \equiv K_{S,RP}$	G3	$Belief \quad Rule$
(13):	$RP\| \equiv S \Rightarrow K_{S,RP}$		$A3$
(14):	$RP\| \equiv K_{S,RP}$	G2	$Jurisdiction \quad Rule$
(15):	$S \triangleleft \{\{N_{RP}, K_{S,RP}\}_{sk_{RP}}\}_{pk_S}$		$M2$
(16):	$S\| \equiv pk_S \mapsto S$		$A1$
(17):	(15), (16) $S \triangleleft \{N_{RP}, K_{S,RP}\}_{sk_{RP}}$		$Seeing \quad Rule$
(18):	$S \triangleleft \{N_{RP}, K_{S,RP}\}_{sk_{RP}}$		MP
(19):	$S\| \equiv pk_{RP} \mapsto RP$		$A1$
(20):	$S\| \equiv RP\| \equiv K_{S,RP}$	G1	$Same \quad Proofs \quad [6-12]$

- Subjects and RPs communication.

4.4.2.1 Security Analysis

In the identity generation process, the protection of the identity information for each subject becomes more critical than traditional IdM systems. By following the same technique used to generate the Bitcoin addresses, the digital identity identifiers in the BIMSIT framework are globally unique to some extent. The double keypairs as credentials of each identity enhance the resilience and robustness of our identity information management. Since the offline keypairs are stored in the offline hardware, they are never exposed to the online environment and runtime execution. The offline key pairs also could be used to revoke online keypairs or identities, while online keypairs are used for updating other information related to the corresponding identity.

Resistance against Impersonation and Cloning Attack. Suppose that the attacker Bob captures a device D_{camera} owned by Alice. He could extract the device-related identity information like online key pair in the memory chip. However, he is not able to replace the online key pair with a newly created one due to the binding signature defined in equation 4.14. Besides, the signature involves the self-examination attributes which is related to the device. By such, the BIMSIT also could prevent cloning attacks of the identity. Unless the mnemonic code words are compromised, it is not likely to impersonate or clone subject identities. Another type of threats can be generated by internal adversaries if they get lost devices without any protection (e.g., a badge for your lock) and impersonate the stored identity. Identities in the hierarchical structure are isolated naturally. Since each subject could generate the identity for every service, one compromised partial identity could not be able to affect other services.

Resistance to External Communication Attacks. In addition to the resilience and robustness of the identity information, we use the DHT-based database (imagine it is a hash table) to store the online identity information set:

$$< key, value >:=< \mathcal{H}_{HASH160}(IDS_o), enc(IDS_o) > \tag{4.17}$$

Therefore, for the subject and DHT peers communication, encrypting of the identity information set ensures confidentiality. Even if someone obtains the identity information, they cannot decrypt it. Moreover, the hash value as the storage key guarantees integrity in that no one, except the owner, could modify the identity set. Besides, only the owner of a specific private key could hold the encrypted secret keys, which makes the authenticity possible. Finally, the distributed network storage ensures the availability of the identity set, which means the identity set is always online. For the communication between subjects/RPs and blockchain IdP, registration, update, and revocation data is plain text and has the signature and hash value to ensure integrity and authenticity. As for the lookup data, we assume each subject has their own trusted blockchain IdP since the blockchain data verified by all peers, is likely stored on their home computer; therefore, the lookup belongs to internal communication and could be considered to be secured. The last type of communication between subjects and RPs has been proved to be

correct in the previous BAN logic method. Therefore, we can conclude that our BIMSIT satisfy the basic security requirements.

Resistance to Single Point of Failure and Phishing Attacks. Our blockchain-based IdP is robust and resilient in that it could eliminate some critical vulnerabilities and defend attacks. Firstly, in the BIMSIT framework, each node can play the role of identity providers as long as the node synchronizes the identity blockchain ledger, which eliminates the single point of failure. Secondly, the BIMSIT framework takes advantage of the ownership between subjects and relying parties to bring the authentication of relying parties into identity management systems. As a result, relying parties are verifiable and trusted, which alleviates phishing attacks. Like the Bitcoin blockchain, our blockchain-based IdP also suffers from the 51% attacks which should be taken into account in real-world applications.

4.4.2.2 Privacy Analysis

Using the blockchain technology to build digital identity providers makes the identity provision more reliable and trusted in that everyone could synchronize the blockchain identity ledger and have a local copy as a trusted blockchain-based identity provider. The distributed blockchain-based IdP is no longer controlled by third parties, which extremely dissipates privacy consideration of private information being utilized by unauthorized parties. Moreover, the detachment of the root identity and partial identity is beneficial to achieve anonymity. Under the hierarchical identity structure in the BIMSIT framework, users also could use partial identities rather the root identity to access services in an anonymity way. This is particularly useful when service providers do not have the Know-Your-Customer (KYC) requirement. All Bitcoin transactions are public and traceable, which renders the Bitcoin system pseudonymous instead of anonymous. If one address is linked to an identity, all related transactions are exposed to others. Compared with the pseudonymity of Bitcoin, our customized transactions are only used for basic identity operations, and other interactions with relying parties are off the chain and cannot be tracked. Besides, subjects can generate as many as partial identities for different application scenarios, which significantly improve the privacy of users.

4.5 Summary

In this chapter, we introduced the blockchain-based identity management system for IoT (BIMSIT) in distributed and trustless peer-to-peer networks. It mainly consists of the system and threat models, the hierarchical identity information management, and the blockchain-based identity provider. Relying on our proposed system and threat models, we also conducted security and privacy analysis to prove that our framework is secured from theoretical perspectives. In detail, we firstly defined some basic concepts (namely, IoT entities, subjects, things, and digital identities) in system model and elaborated the relationship between these concepts and three IdM stakeholders (namely, subjects, relying parties, and identity providers) as shown in Figure 1.1. Then, we clarified the capabilities of attack-

ers in the threat model to help us do security and privacy analysis in section 4.4. Based on our proposed system model, we adopted the cryptographic algorithms used in the bitcoin hierarchical deterministic wallet to generate hierarchical identity trees in our IdM system. We also presented the blockchain-based IdP, which creates a distributed public ledger used for managing root identity transactions of subjects from registration, update, revocation, and lookup in trustless peer-to-peer networks. At last, we formally verified our BIMSIT using the BAN logic and analyzed security and privacy properties from the theoretical perspective.

The proposed identity system model and hierarchical identity trees give all IoT entities verifiable identities. More importantly, the proposed distributed public ledger as the IdP eliminates unnecessary intermediaries in trustless environments. Therefore, our proposed identity management framework is a paradigm-shift by which any user can create an identity hierarchy which is exclusively managed by self-sovereign owners without relying on third-party identity providers (e.g., Google, Facebook). Compared to centralized and federated identity management systems, the proposed identity framework removes single point failure (centralized servers). Although the blockchain-based identity provider can eliminate privacy concerns caused by traditional centralized IdPs and the hierarchical identity structure can provide anonymity services to some extent, the public blockchains still could expose identity information. In other words, the endorsement from blockchain root identities is required by partial identities from time to time, which increases the risk of being compromised. Therefore, complete privacy has to rely on other privacy-preserving solutions such as multi-party computation or zero-knowledge proof [Aug+17b; Aug+17a; HP16]. Besides, the goal of establishing identities is to define access privileges to communicate without barriers. In other words, supporting access control framework should be closely tied to the identity management framework.

CHAPTER 5

Proposed IoT Access Control Framework

Contents

5.1 Introduction

Once every entity in IoT has a digital identity as proposed by the BIMSIT framework, we develop a decentralized IoT access control framework which regulates access permissions during interactions between IoT entities. IoT-based systems and services interact with each other by relying on different levels of trust relationships, and consequently, require ultimate security solutions to protect information and resources. With the wide use of devices and IoT communication protocols, we are experiencing grand challenges to secure and protect IoT services due to the significant increase in the attack surface [Nas+09]. On the one hand, the broad spectrum of IoT makes designing security mechanisms (e.g., access control) challenging. Moreover, current centralized access control systems, which are traditionally built around centralized third-party identity providers raise security and privacy concerns such as the single point of failure and user-privacy leaks. On the other hand, various devices with different computing capabilities also pose

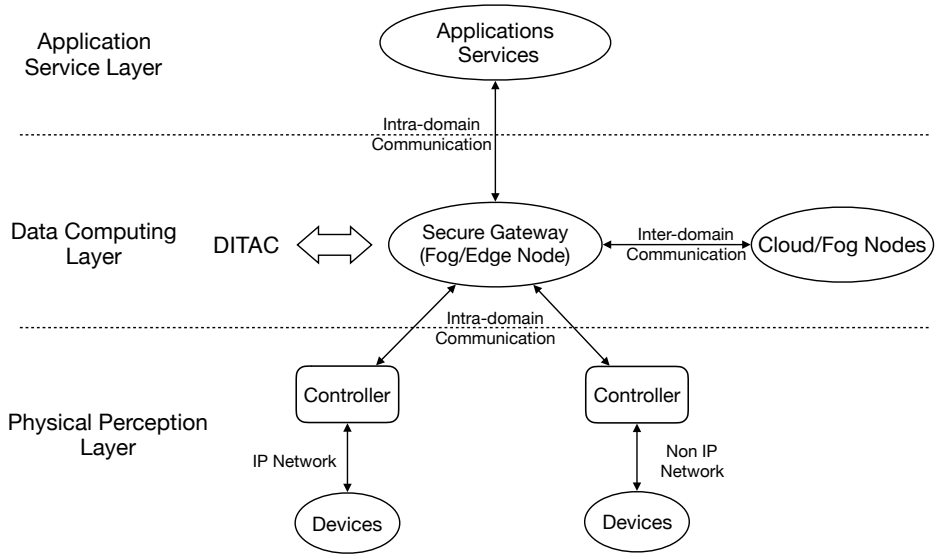


Figure 5.1 – IoT framework, including our decentralized IoT access control framework

a significant challenge on preventing malicious users from breaching system and stealing sensitive information of these devices at runtime.

Figure 5.1 shows the framework used to build trustworthy and resilient IoT applications in peer-to-peer networks of various fog nodes. It consists of three layers: physical perception layer, data computing layer, and applications services layer. It also guides the security development of IoT services by deploying access controls on gateways.

The physical perception layer includes key entities such as sensors for capturing states of the physical world and representing them as services in the digital world. The key entities also include actuators to act on the physical world and change its states [Suo+12; HFH15]. The physical perception layer focuses on exchanged information about devices. In addition, environmental conditions pass through devices to identify or sense the physical world. At this level, cyber attacks target local controllers, sensors, actuators, and exchanged information. In our IoT access control framework, physical devices or things are subject-centric since they are associated with their owners. Consequently, we adopt a user-centric approach to built security mechanisms driven by users/subjects without relying on third parties.

The data computing layer comprises two types of computing nodes: Fog/Edge computing nodes and Cloud computing nodes. We assume that each user/subject owns and manages one or many (secure) gateways (i.e., Fog/Edge nodes) used for enabling their security policies and access controls to services provided by entities. The communication network among devices and Fog/Edge/Cloud nodes includes mobile communication networks, wireless sensor networks, network infrastructures, and communication protocols [HFH15]. Network security and management play an essential role to defend against cyber-attacks targeting firewalls, routers, protocols, and personal information. Attacks on the communication layer, have impacts on reputation, safety, energy, control, and time of

the IoT framework. Network mitigation mechanisms include authentication, anti-DoS, encryption, packet filtering, congestion control, anti-jamming, intrusion detection, and behavior analysis. Compared to Fog/Edge computing nodes, Cloud nodes, which are usually operated by Cloud service providers, have massive computational power. The intensive application data, which could not be handled in Fog/Edge nodes due to limited computing capabilities, is processed in Cloud nodes representing Cloud data centers. In Cloud nodes, cyber attacks could target personal and confidential information. Cyber attacks' impact includes people safety, money losses, and vital information leakage. Protection mechanisms include encryption, authentication, session identifiers, intrusion detection, selective disclosure, data distortion, and behavior analysis.

The application service layer provides personalized business and IoT-based services according to user needs [HFH15; MW11]. Access to these services is possible through mobile technology (e.g., cellphone, mobile applications) and smart appliances. In this layer, data sharing is an important characteristic, and consequently, application security must address data privacy, access controls, and information leaks. Attacks' impacts can be unauthorized accesses to intellectual properties, disclosure of critical business plans, money loss, and damaging business reputation. Some mitigation mechanisms include encryption, authentication, and anomaly behavior analysis of applications and services.

In order to deal with these challenges, we present the Decentralized IoT Access Control framework (DITAC) for edge computing nodes enabled it with an autonomic threat detection mechanism. By such, users can manage their security policies in trustless IoT environments and detect threats of the lower layer. Specifically, our DITAC achieves the following contributions:

- **Universally Social Model for IoT**, which takes advantage of three types of relationships (S2S, S2T, T2T) to unify all IoT entities from people, companies, organizations to devices, applications, and services.
- **User-centric Access Control**, which makes use of capability-like relationships to build trust in trustless IoT environments and grants permissions via mapping users of two participants.
- **Autonomic Threat Detection at Runtime**, which trains a threat reference model to detect anomalies triggered by any unusual event (e.g., cyber-attacks) in physical device layer, and provides the required intelligence to access control framework so that the upper layers will deal with these security concerns.

The remaining sections are organized as follows. Section 2 complements some definitions, such as users and policies in access control systems, based on previous definitions in our IdM framework. Then, we model access control systems using these definitions and explain different parties in the access control model. Section 3 presents the decentralized IoT access control framework. Section 4 introduces the autonomic threat detection mechanism for our IoT access control framework.

5.2 IoT Access Control Model

In the identity system model presented in chapter 4, we introduced fundamental concepts to abstract things and entities in the IoT. For instance, a subject denotes individuals or collectives who have a one-to-one correspondence with the juridical person in social life. Similarly, we introduce the access control system model by which subjects refer to digital entities as active processes in computer systems and carrying the will of juridical persons and initiating access requests to services. As defined in previous *Definition* 3, things are composed of physical things and virtual things. In order to differentiate these concepts, we use (smart) objects to denote physical things in this chapter. Based on the identity system model, we introduce a couple of definitions and assumptions before introducing the access control model and the decentralized IoT access control framework:

Assumption 2 *All subjects (i.e., individuals or collectives) could own things as assets (asset ownership).*

Definition 5 *(User): A user refers to one partial identity of subjects in a specific context, which comprises an identifier, credentials, and attributes. For instance, Alice, an employee, is a user defined by its unique identity with responsibilities (e.g., roles) related to her company. For the sake of simplicity, users and their identities are interchangeable unless we explicitly.*

Definition 6 *(Service Provider): A service provider refers to the combination between a smart object and the specific set of service interfaces, residing in the smart object. It provides a set of services to users to interact with physical things and/or virtual resources.*

Definition 7 *(Policy): Policies are a set of rules enabled by owners of service providers to regulate authorized accesses to their services.*

Hence, we define the IoT access control system model which comprises four components, namely, users (U), smart objects (T), services (S) and Policies (P) as shown in Figure 5.2. A user (U_a) firstly needs to send an access request to a smart object (T). The access request designates the requested service from the service provider. Then, T checks the owner of itself. If U_a is the owner of T, T_a grants full access to smart object services. Otherwise, T_b has to check the policy(P_b) or inquiry the owner U_b to make a decision regarding the access request.

5.3 Decentralized IoT Access Control Framework

Based on the proposed access control model, we elaborate our decentralized IoT access control framework from four aspects: subject management, relationship management, object management, and access control mechanisms.

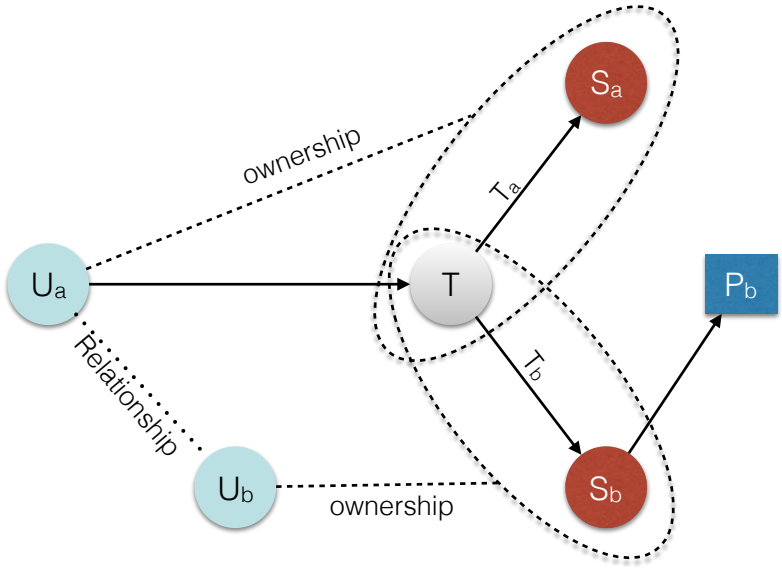


Figure 5.2 – Access control system model abstraction

5.3.1 Subject Management

As described in our proposed access control model, each subject could have many users or identities which are used in different contexts. In our previous chapter 4, we proposed the hierarchical identity structure to manage all identities of a subject as illustrated in Figure 5.3. Specifically, for any member of *Subjects*, that is to say, collective subject or individual subject, we introduced the hierarchical structure of identities. The hierarchical identity structure reflects various relations such as organizational, ownership or security authority relations, among sibling identities or parent-child identities. Besides, we use the **Root Identity** (RId) to represent the source identity from which a collective subject or an individual subject generates one or more **Partial Identities** (PId) each of which identifies a domain or sub-domains.

A domain is a set of computational and storage resources that are available to a subject, services that are working under the subject authority or, in general, things that are managed by the same subject. Domains can be separated by logical boundaries and architected in a hierarchical manner such as sub-trees in the hierarchical identity structure. The relationship between different domains dictates how two or more subjects can communicate or access resources, services or things of each other. As a result, the entire hierarchical identity structure can be seen as a tree in which each node is expressed by its digital identity and the domain or sub-domain under its control.

As mentioned in the previous chapter 4, identity is composed of identifier, credentials, and attributes as depicted in Figure 5.3. In order to manage the identifiers and credentials under the same subject, we proposed the hierarchical identifier and credentials backup solution based on the Bitcoin Improvement Protocol 32 ($BIP32$). All the identifiers and credentials (more specifically, offline keypairs) of the hierarchical identities could be generated and restored from a random seed. When a subject wants to add an identity subbranch of a domain,

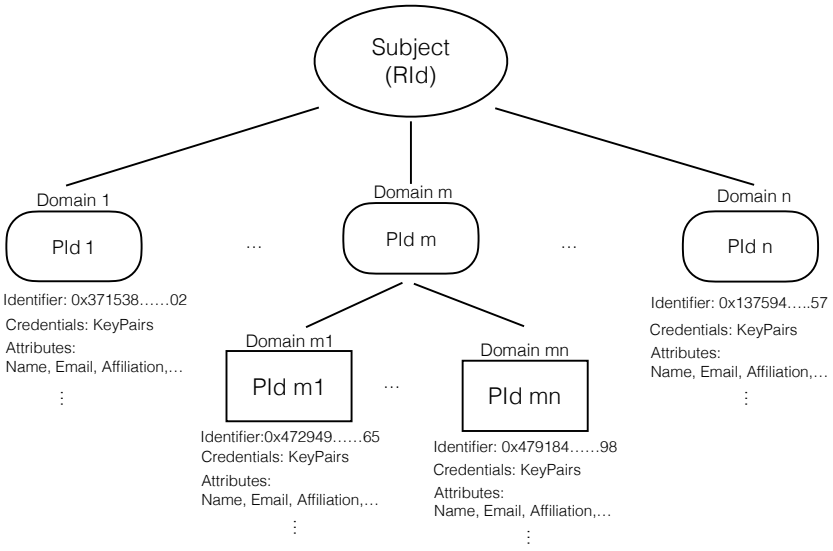


Figure 5.3 – Hierarchical identity structure

an index number will be calculated according to the position of the subbranch and used for generating the corresponding offline key pair of the identity. Besides, we adopted the double key pairs: the online and the offline key pairs to increase the reliability of the system. For a subject identity, its identifier is obtained from the hash value of its offline public key by using the double hash algorithm $HASH160$. Its credential is a set of claims made by a subject about its identity such as public and private key pairs. Therefore, we can describe identity as follows:

$$IDENTITY := identifier < offline\_pbKeyhash > credential < keypairs > attributes \; [< key = value >]^{+}$$

$$IDENTIFIER := HASH160(offline\_public\_key)$$

$$CREDENTIALS := offline\_keypair, online\_keypair$$

Also, we define the role concept [McC05] in our access control model. A user, defined by its partial identity in a specific context, may have one or more roles. Instead of being uniquely associated with one subject identity, a role defines a specific domain and is intended to be assumable by any identity who needs to access resources or services. Therefore, we have the following definitions:

Definition 8 *(Domains): A domain is a set of services under the authority of a specific user (or, partial identity) dominated by roles.*

Definition 9 *(Roles): A role is a collection of predefined services in a specific system.*

A subject, through his/her root identity, provides contextual access to his/her owned smart objects (i.e., resources and services) by creating different domains. Each domain designates a contextual access control authority, consisting of roles and a corresponding identity in a given context. By such, the subject root identity creates its structural or organizational hierarchy of different identities each of

which is assigned one or more roles. The hierarchy of identities can be seen as a refinement of access control permissions where the role (and the access control privileges) at the level n is more role restricted from the role at the level $n+1$. As a result, the root identity organizes access controls of his/her identities to smart objects under his/her security authority.

After generating identity hierarchical identity tree, subjects have to construct identity transactions using their root identities and post corresponding transactions to blockchain networks for reaching identity consensus. Then, subjects can take advantage of root identity transactions mined in blockchain networks to authenticate themselves without relying on unnecessary third-party identity providers and to manage all IoT things falling into the same subject through creating identities for these things. The related concepts can be expressed as described:

$DOMAINS := domain < domain\_name > IDENTITY\ dominance\ ROLES$

$USERS := user < user\_name > roles\ [< role\_set >]^{+}$

$ROLES := role < role\_name > services\ [< service\_set >]^{+}$

5.3.2 Relationship Management

In order to ensure access security between subjects, we introduce relationships to grant permissions via mapping users of two participants in trustless IoT environments. Here, we define:

Definition 10 *(Relationship): In sociology, relationships define how two or more individuals are connected, while, in our framework, we think of in particular relationships as a binary relation between two subjects ($\mathcal{S}$). Therefore, given a binary relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ and $a, b \in \mathcal{S}$, we write $a\mathcal{R}b \iff (a, b) \in \mathcal{R}$. Also, we use the collective of subjects to represent non-binary relationships in our framework so that all non-binary relationships could be converted to the binary relationship. In order to resolve conflicts, we only consider the direct relationships between two subjects without transitive ones.*

Two subjects could declare a relationship between them through a negotiation process. Technically, a relationship can be expressed by the relationship declaration structure comprising the header (version, identifier, label, timestamp, valid date, data field hash value, signatures) and payload (capability connectors and data field):

- the *label* refers to a semantic description of the relationship which both subjects are willing to establish such as patient-doctor, university-graduate, host-guest. The labels could be predefined, supported by service providers, or negotiated between two parties;
- the *data field* could store all the related provable documents like driver license, birth certificate or passport issued by issuers in the relationship;
- the *capability connectors*, in essence, are one-to-one users (namely, the combination of partial identity identifier and role) mapping list in a specific context used for communication and access permission management.

Hence, relationship related concepts can be formally described as follows:

$RELATIONSHIP := HEADER \quad PAYLOAD$

$HEADER := id < relationshipIdentifer > label < relationshipLabel > timestamp$
$< timestamp > version < version > subject1 < subject1RootIdentifier >$
$[subject2 < subject2RootIdentifier >] created < cDate > expired < eDate >$
$sig < relationshipSignature > Integrity < hashval >$

$PAYLOAD :=< data\_field > [< key = value >]^{+} CAPABILITY$

$CAPABILITY := [< subject1PartialIdentifier : role > < subject2PartialIdentifier : role >]$

5.3.3 Object Management

According to our access control model, smart objects are accessed and controlled through their services (i.e., APIs). In order to complete the access control scheme, we define the object management to handle the lifecycle of smart objects, which includes object registration, object service provision, and object disposal.

- Smart Object Registration. When smart objects are connected to the IoT-based networks through sensors or actuators, subjects with root identities need to register them into their authority domains to render them identifiable and accessible. The registration operation creates object profiles, assigns them to the objects' owner, and specifies their access points (i.e., mount point or *URL*).
- Smart Object Services Deployment and Security Policies Configurations. After registering smart objects, subjects assign them access policies to control the execution of their services and decide who could access them based on roles and relationships.
- Smart Object Disposal. When subjects decide to remove smart objects from the DITAC framework, the dispose operation reverses the registration operation and drop all related roles, relationships, policies, and rules.

As a result, we give the grammar-based language of object management and its access control policy syntax as follows:

$OPERATIONS : [register|dispose]\ OBJECT^{+}$

$OBJECT := name < object\_urn > identifier < partialIDIdentifier > access < url >$
$owner < rootIDIdentifier > SERVICE^{+}$

$SERVICE := name < service\_name > url < url\_value >$
$input \quad INPUT^{*} \ output \quad OUTPUT^{*}$

$POLICY := RULE^{+}$

$RULE := on\ [< domain\_name > | < relationshipIdentifer >]^{+}\ access\ things[< object\_name >$
$| < service\_name >]\ [accepted|denied] \quad [where \quad SECURITY\_CONTEXT]$

$SECURITY\_CONTEXT := [< role\_attribute > LOGIC\_OPERATOR < role\_value > |$
$< relationship\_attr > LOGIC\_OPERATOR < rel\_value > |$
$< context\_attr > LOGIC\_OPERATOR < ctx\_value >]^{+}$

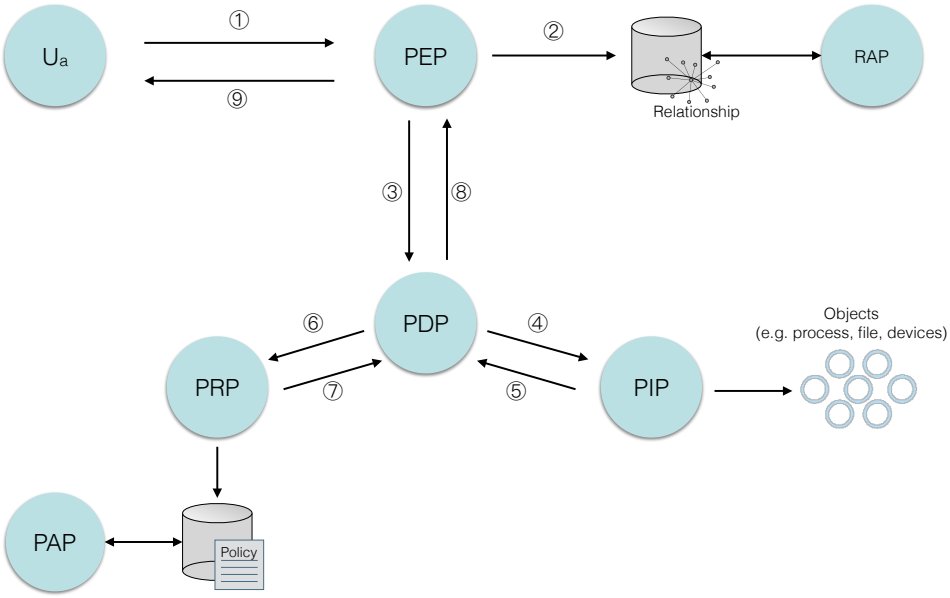


Figure 5.4 – The proposed decentralized access control mechanism for IoT

5.3.4 Access Control Mechanism

The DITAC framework makes possible to enable secure access controls between two different users (or digital identities), belonging to the same security authority (i.e., same subject), or belonging to two different security authorities each of which belongs to a different subject.

Suppose a user under the authority of **Subject a** wants to access a service provided by a smart object owned by **Subject b**. To this end, **Subject a** and **Subject b** need to establish a trust relationship and designate the corresponding users in the capability connector field via the Relationship Administration Point (RAP). When **Subject a** discovers or decides to access the service provided by **Subject b**, **Subject a** sends a "service request" message which is signed by the online private key of **Subject a**. The "service request" message also includes information about the user from **Subject a**. If **Subject b** agrees to grant access, a corresponding user under the authority of **Subject b** is chosen to configure the access control policy that grants proper permissions. In case both users belong to the same subject, the relationship between these two users (via their identities) still required before accessing their services. In this way, inside attacks could be isolated in specific domain even if one identity is compromised. Figure 5.4 shows how the entire access control mechanism works, and this mechanism is fully described as follows:

- (1) a user (U_a) of **Subject a** sends the access request to the Policy Enforcement Point (PEP) which could reside in service providers or on the access control gateways of **Subject b**;
- (2) the PEP authenticates the U_a in order to map it to a known user of **Subject b** according to the eventual relationship (e.g., predefined friend or family member) between the two subjects;
- (3) the PEP delivers the request to the Policy Decision Point (PDP) for making the decision;

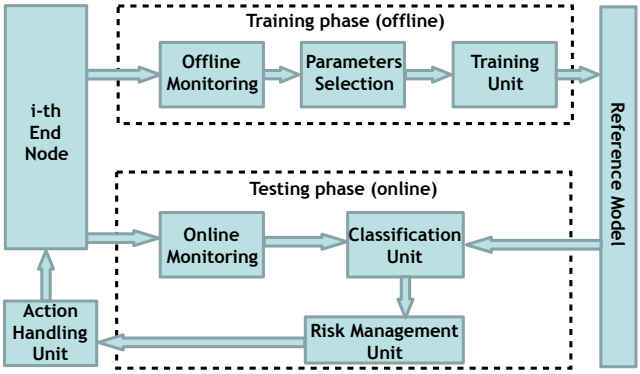


Figure 5.5 – Enabling IoT security at the physical perception layer

- (4) - (5) the PDP sends a query to the Policy Information Point (PIP) to retrieve the corresponding security context including the corresponding user (U_b) and the requested object information;
- (6) - (7) the PDP queries the Policy Retrieval Point (PRP) to retrieve the access control policies related to the corresponding domain;
- (8) the PDP evaluates the access request by running the security context against the policies and then sends the result to the PEP;
- (9) the PEP decides whether or not to grant the access permission of the service to **Subject a** according to the returned result.

Once the access control is granted, an access session, which designates the context under which a subject accesses resources and services, is monitored by our autonomic threat detection system to detect abnormal behaviors and consequently apply appropriate risk analysis and mitigation. By such, our proposed access control framework is not static and remains aware of any changes that may threaten the IoT framework integrity and usability.

5.4 Enabling Access Control with Autonomic Threat Detection

In order to protect operations of smart objects, gateways, or end nodes and support our access control at runtime against security threats, we developed a continuous monitoring and performing anomaly behavior analysis system to monitor the end nodes' operations [PH16]. We augment the monitoring capabilities by focusing on end-devices (smart objects and connected devices) as they become easy targets for cyber-physical attackers. It is worth noting that the main problem is that a computational device (e.g., a microcontroller) can be attached to the IoT realm, following closely the behavior of a given node, representing a threat to the entire IoT-based system. The basic idea is to create a threat reference model for each smart object that describes its normal behavior. Figure 5.5 shows our autonomic threat detection approach to handle security issues at the IoT physical perception layer via:

- a training phase to create the reference model, and
- a testing phase to perform behavior analysis with respect to this model.

Consequently, the trained threat reference model can be used to detect any abnormal behavior of users during the access to resources and services

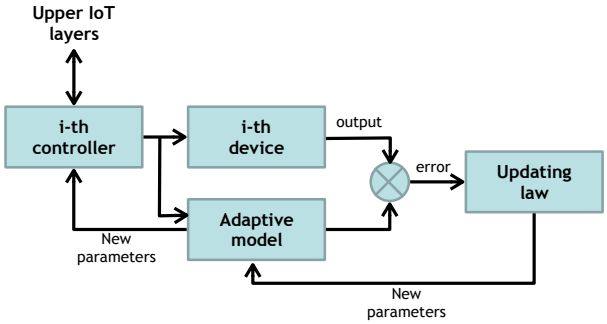


Figure 5.6 – Decentralized control scheme for the *i-th* node

5.4.1 Training Phase

The training phase aims to build a machine learning based model that characterizes the normal behavior of smart objects or any node at the edge of IoT. This phase is conducted offline. In what follows, we describe each step in the training phase based on Figure 5.5.

- **Offline monitoring.** This module is in charge of continuously monitoring the performance of smart objects and devices. All parameters are collected here depending on the type of objects or devices being inspected. Usually, the offline monitoring is performed by a local controller deployed on a gateway or an isolated computer.
- **Parameters selection.** A fundamental step is to choose the parameters that better describe the smart object and device behavior accurately. For instance, if the device is a direct current electric motor (namely, DC motor), we are interested in collecting information about motor speed and current consumption.
- **Training unit.** The training unit is responsible for building the model of the device being inspected. In this case, we can lean on a predefined model (e.g., given by the provider), or we can obtain a parametric model using machine learning techniques such as artificial neural networks.

5.4.2 Reference Model

Our proposed scheme is intended to work with smart objects and devices that can perform self-configuration, self-healing (automatic discovery correction of faults), self-optimization (automatic monitoring and control of resources), self-protection (proactive identification and protection from threats), self-regulation (maintain steady state without external control), and self-management (manage itself without external intervention). To achieve these goals, we propose the decentralized controller scheme shown in Figure 5.6, which represents an arbitrarily connected smart object. The decentralized control scheme is a robust solution to address the monitoring of large-scale complex systems of objects such as transportation systems, electric power grids, and communication networks, to mention a few [JNL11]. In what follows, we describe the functional building blocks of our proposed approach applied to the i-th node.

5.4.2.1 i-th Device

A device in the IoT performs tasks such as managing and processing data before sending them to the upper layers. Devices are at the edge of IoT and mainly interact with the physical world. To enable the control of device behaviors, we need to build a reference model. However, it is difficult to obtain a model due to the high complexity of the device and its physical restrictions. For example, the device is distributed on large geographical locations or have multiple instances (i-th device). In the case of a plant, we consider that the model is described as a perturbed affine nonlinear system that can be represented by Equation (5.1).

$$\begin{aligned} \dot{x}_i &= f_i(x_i) + g_i(x_i)u_i + \xi_i(t) \\ y_i &= h_i(t) + v_i(t) \end{aligned} \tag{5.1}$$

where $x \in \mathbb{R}^n$. Let $\eta_i(t)$ be a known/unknown perturbation that can be caused by a cyber-attack that affects the i-th node, then the perturbed output $\tilde{y}_i(t)$ is defined as

$$\tilde{y}_i(t) = y_i(t) + \eta_i(t) \tag{5.2}$$

Equation (5.2) can be simplified using Equation (5.1) to obtain Equation (5.3).

$$\tilde{y}_i(t) = h_i(t) + v_i(t) + \eta_i(t) \tag{5.3}$$

Notice that, in the model of the output represented in Equation (5.3), both the internal and external perturbations are included. This is to represent the dynamical behavior of the system and cyber-attacks respectively. In addition, for a complex system, it is difficult to discriminate when an abnormal behavior is caused by its dynamic or by an external command. This issue will be addressed in the following subsection.

5.4.2.2 Adaptive Model

The decentralized adaptive system takes into account the dynamics of the device considering two approaches for the fault detection task:

- internal fault: due to the dynamical response of the device which is affected by its internal operation and by perturbation signals that are inherent at the rated conditions;
- external fault: due to the perturbation caused by cyber-attacks.

The adaptive model must be able to distinguish from an internal/external fault, and capable of restoring the *i-th* node to its healthy operative condition with a minimal or no perturbation to the whole system. For the model of the *i-th* device described by Equation (5.1), we propose to use a decentralized recurrent high-order neural network model (D-RHONN) whose properties have been proved in [BSL07] as an excellent adaptive identifier for nonlinear plants. The structure of the D-RHONN is described by Equation (5.4).

$$\begin{aligned} \dot{z}_{i1} &= z_{i2} \\ \dot{z}_{i2} &= -a_{i2}z_{i2} + w_{i21}s_i(x_{i1}) + w_{i22}s_i(x_{i2}) + \overline{w}_{i1}u_i \end{aligned} \tag{5.4}$$

where z_i is the neural network state; $a_i > 0$ are parameters chosen to stabilize the neural network behavior; w_i are the synaptic weights; u_i is the controller signal which drives the *i-th* node; $\overline{w}_{i1}$ are fixed parameters known a priori; and s_i is the sigmoid function that constitutes the activation functions of the neural network and is defined by Equation (5.5).

$$s_i(x) = \frac{a_i}{1 + e^{(-\beta_i x)}} - \gamma_i \tag{5.5}$$

5.4.2.3 Updating Law

The outputs of the device and its corresponding model are compared to generate an error signal. Depending on the nature of the fault, two types of errors are considered: dynamic errors ed, which occurs due to the dynamical behavior of the device (i.e., plant); non-dynamical error $\sim ed$, due to a cyber-attack performed by external entities. The composition of the error et can be described by Equation (5.6).

$$et = ed + \sim ed \tag{5.6}$$

with $ed = y_i - \check{y}_i$ as the model of the dynamical error according to Figure 5.6. The total error drives a learning law block, which generates new parameters to the adaptive model in a closed loop architecture that converges to the device output once that $et = 0$ or very close to it.

5.4.2.4 Controller

The controller is driven by three building blocks:

- the adaptive model, which is selected as a priori (training phase), provides the parameters to the controller to assess threats and decide on operations;
- the control law, whose structure describes the desired operation of the device, this law is restricted to the performance policies that the tracking block dictates to the control law;
- the gateway, which is the observer of the behavior, and is responsible for propagating the behavior information of the inspected objects or devices to the upper layers.

Let $\eta_i(t)$ denote an additive unknown perturbation that models an attack to the *i-th* node. The output of the perturbed system is $\tilde{y}_i = y_i + \eta_i(t)$, then the attacked model of the system is represented as Equation (5.7)

$$\begin{aligned} \dot{x}_i &= f_i(x_i) + g_i(x_i)u_i + \xi_i(t) \\ \tilde{y}_i(t) &= h_i(t) + v_i(t) + \eta_i(t) \end{aligned} \tag{5.7}$$

The output $\tilde{y}_i(t)$ is compared with the output of the adaptive model $\check{y}_i(t)$ to generate the total error according to Equation (5.8)

$$e_T = e_d + \tilde{y}_i - \check{y}_i \tag{5.8}$$

Therefore, from Equation (5.9), the error due to a cyber-attack is represented by

$$\sim e_d = e_T + \check{y}_i - y_i \tag{5.9}$$

It is important to notice from Equation (5.9) that it is imperative to analyze how to distinguish the source of the perturbation in the dynamics of the device, in other words: to what extent a cyber-attack contributes to the global behavior of the perturbation? at which point the transient state is due to the natural dynamic response of the system? It is clear that the dynamical and non-dynamical errors are bounded by

$$||\sim e_d|| = ||e_T|| + ||\check{y}_i - y_i|| \tag{5.10}$$

Then, a restrict constrain must be imposed on the non-dynamical error, which is bounded by the total error given by Equation (12)

$$||\sim e_d|| \geq ||e_T|| \tag{5.11}$$

A big issue to deal with is the capability of the methodology to distinguish a false positive that mistakes a cyber-attack from a dynamical transient. This problem can be solved if we guarantee that the adaptive model converges fast enough to the device reference model (i.e., the plant reference model in previous equations).

5.4.3 Testing Phase

Once we have obtained the reference model at the training phase, the next step is to verify if the model successfully discovers the node that is sending the information about the device behavior to the upper layers.

- **Online Monitoring.** This module is in charge of continuously monitoring the performance of the device at the physical perception layer. It works similarly as in the offline; however, it filters the required parameters to send only required information to the classification unit.
- **Classification Unit.** This unit is the heart of our autonomous approach. It performs the identification based on the model obtained offline. This unit can be used to authenticate the object or device that is sending information from a given location (or IP).

The parameters are obtained online from online monitoring unit and then compared against the reference model. It is clear that for a dynamic system (e.g., a DC motor), subject to external cyber-physical disturbances, the reference model may be not enough to identify the inspected node accurately, and hence we need to verify the error. Given e_T in Equation (5.8), we can compute

$$\mathcal{L}_2|e_T| = \sqrt{\frac{1}{T}\int {e_T}^2 dt} = k \tag{5.12}$$

where we can statistically establish a threshold for k for a time of interest T [Mon09], such that if the threshold is bypassed, the end node is either under attack or a malicious device is trying to send information on behalf of the legitimate one. Ten samples of k are used to find the control limits for normal operation [Wan+12]. Once all the samples are taken, the mean value is calculated as the control limits (CL). Then the Upper Control Limit ($UCL = \overline{x} + \alpha\sigma$) and the Lower Control Limit ($LCL = \overline{x} - \alpha\sigma$) for the normal behavior is computed, where $\overline{x}$ is the mean

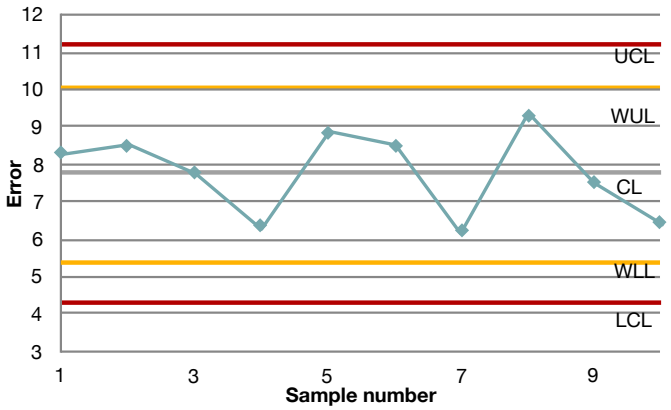


Figure 5.7 – Control chart for normal error

value, $\overline{\sigma}$ is the standard deviation and α is a sensibility level. For normal control limits, we assume $\alpha = 3$. However, we can establish warning upper and lower limits (WUL and WLL, respectively) at $\alpha = 2$. Figure 5.7 shows the control chart for the normal behavior of errors.

5.5 Summary

In this chapter, we introduced a decentralized IoT access control (DITAC) framework with autonomic threat detection. Specifically, based on the blockchain-based identity framework proposed in the previous chapter, subjects can manage all IoT things falling into the same subject domain by creating hierarchical identities for these things. The blockchain based identity management framework eliminates unnecessary centralized identity providers through creating blockchain identities, in which all users and service providers follow the identity consensus and hence could verify identities instead of blindly trusting in some big third-party identity providers. However, individuals still do not trust (or know) each other, even if they could verify the real identities. In DITAC, we make use of capability-like relationships to build trust in trustless IoT environments, which unifies all IoT entities from people, companies, organizations to devices, applications and services. More importantly, relationships allow owners to set user-centric security policies to secure their own devices or services without relying on other unnecessary intermediaries. Before granting access permissions, the owner and accessor of services need to negotiate terms to see if the accessor could meet requirements prescribed by the owner. After that, trusted relationships are established between these two parties and corresponding access permissions can be granted to requesters of services. In order to recognize vulnerabilities in each layer, we also enabled the framework with a threat model that could be used to identify potential attacks in the physical perception layer. In detail, we integrate an autonomic threat detection to generally create a reference model for each smart object that describes its normal behavior and perform behavior analysis at runtime, to detect attack surface and, in particular, abnormal behavior during access control session.

However, the established relationships in DITAC are too primitive to express complex access control scenarios such as multi-subject with transitive relation-

ships for access control delegations. Relationship establishment templates need to be developed to support various application scenarios in IoT environments, such as employer-employee, patient-hospital, and college-alumnus. Besides, the current threat detection mechanism is designed for the physical perception layer and hence detecting attacks by analyzing the behavior of other IoT layers is conducive to the completion of runtime IoT threat detection module. For example, if an attacker collects enough sensors' information (e.g., one day of information), it can launch a replay attack without the need of using the same data set. A potential research direction for such kind of attacks, including 1) Big data analytics to search for patterns in the stored data, and 2) Moving target defense to change configurations, making it extremely difficult for an attacker to collect relevant data.

CHAPTER 6

Implementation and Evaluation

Contents

6.1 Introduction

In previous chapters, we presented the FoCuS infrastructure to build the Internet of Things as social-like, decentralized peer-to-peer networks. The FoCuS has two pillars: the blockchain based identity management framework (BIMSIT) and the decentralized IoT access control framework (DITAC), including the threat detection capabilities. Figure 6.1 illustrates the relationship between previous chapters and the key FoCuS implementation modules mentioned in the FoCuS architecture (Figure 3.5).

- The identity and access control management (IAM) module implement hierarchical identities and access policies management functionalities.
- The blockchain based identity provider (BIdP) module is responsible for managing identity transactions of subjects and Bitcoin blockchain.
- The threat detection module, which supplements and supports the designed IoT access control framework, ensures runtime security of the FoCuS infrastructure.

Besides these key functional modules, there are another two modules mentioned in our previous architecture: the dashboard module and the device management module. The former is Web-based Graphical User Interface (GUI) through

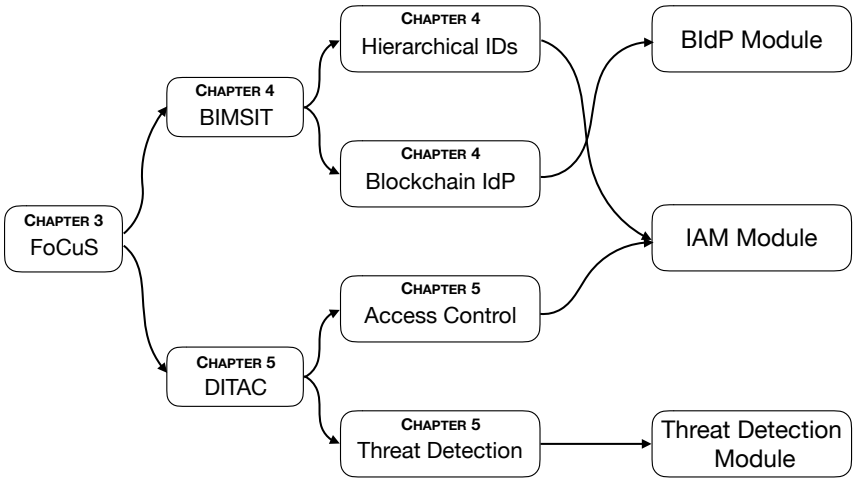


Figure 6.1 – Relationship between chapters and implementation modules

which users can monitor the status of each module and configure the FoCuS through provided public REST APIs. The latter is responsible for managing IoT devices, which involves many non-security concepts such as virtualization and abstraction. In order to demonstrate the feasibility of our contributions, we implement a minimal viable prototype to manage Bitcoin blockchain-based identities and set access control policies with case studies, covering different scenarios. The prototype is modular with event-driven communications. Each module is self-contained and can be later extended towards a fully functional product. Also, we perform analytics and experiments with the autonomic threat detection system through a simulation environment.

This chapter covers the implementation and evaluation of our IoT secure infrastructure – FoCuS, and provides details regarding the IAM module, the BIdP module, the dashboard module, and the threat detection module. Taking into account the previously mentioned four IoT unconventional characteristics (scalability, heterogeneity, dynamic changes, limited resources), we develop the FoCuS prototype global architecture based on the Vert.x [1] framework which provides a toolkit for building event-driven, responsive, resilient, and elastic applications. The advantages of event-driven and non-blocking make our prototype suitable for IoT dynamic environments, in which the designed prototype should be able to handle massive concurrency using limited computing resources responsively. Besides, the Vert.x toolkit for building reactive applications offers various components such as service discovery, circuit breaker, Vert.x configuration to build microservice-based applications. Originated from the service-oriented computing paradigm, microservices could not only define a set of interoperable REST APIs like SOA but benefit from the elastic architecture which makes microservice-based applications more scalable in IoT environments and easy to integrate other sub-projects written in different programming languages. We listed the Vert.x components used in the FoCuS implementation as follows.

- Java Vert.x Core
- Vert.x Web

[1] https://vertx.io/

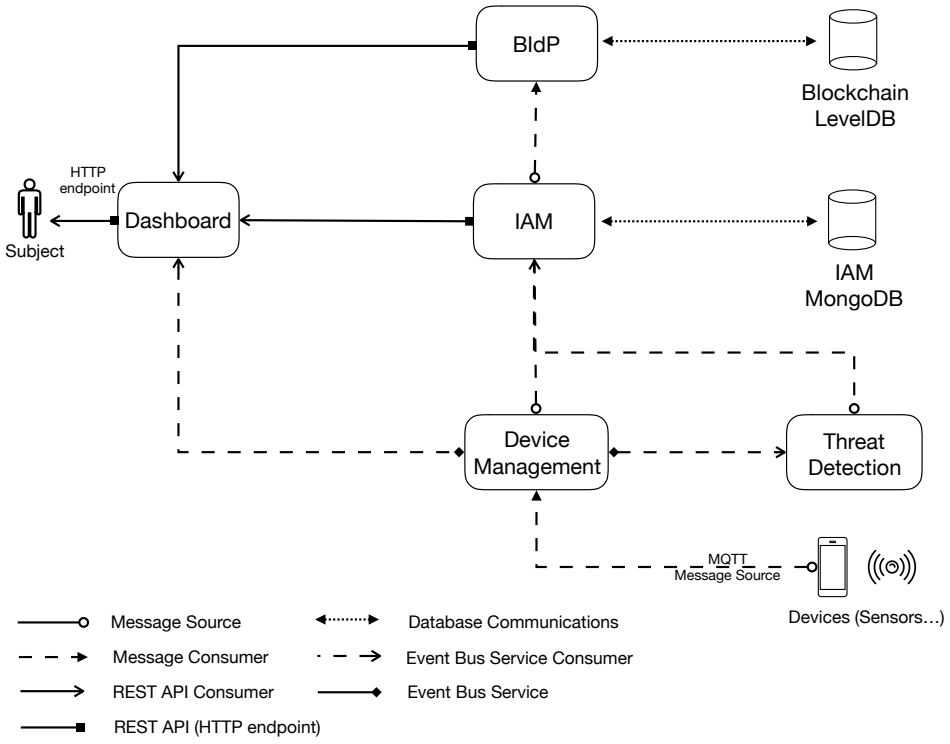


Figure 6.2 – FoCuS microservice architecture

- Event Bus Service
- Microservice Architecture
- MQTT
- Service Discovery
- Circuit Breaker
- Vert.x Config

Figure 6.2 demonstrates the microservice architecture of FoCuS using the Vert.x. More precisely, we use the Java programming language to write the core identity and access control management (IAM) module in FoCuS, and the Maven software management tool to integrate other modules such as the dashboard, blockchain-based identity provider (BIdP), threat detection and device management modules. The IAM module provides REST APIs for the front-end dashboard module while the back-end uses MongoDB to store all identities and access policies information from the IAM module. Therefore, the IAM module also exposes interfaces to the BIdP module. In other words, subjects who need to register their root identities, have to post their identity transactions to the BIdP module through these interfaces.

The BIdP module mainly integrates a modified Bitcoin client (C++ Bitcoin client v0.16, Feb 2018), which can accept our proposed identity transactions. Through the event bus, subjects use the BIdP driver in the IAM module to deliver identity transaction messages to the modified Bitcoin client (BIdP) for registering, updating, or revoking operations. We also develop the monitoring dashboard module to monitor the BIdP nodes in the network. The device management module offers the access point to IoT devices and is responsible for managing data generated by IoT devices through MQTT. Generally, subjects rely on exposed

messages and event services from the device management module to monitor the real-time status, manage the identity of devices, and detect abnormal behavior through threat detection module.

The remaining sections are organized as follows. We firstly illustrate implementation details of main modules in FoCuS. Then, we make use of our implementation to reproduce the case studies presented in previous chapter 3, analyze the performance of our FoCuS from the perspective of scalability, interoperability, mobility, storage, communication and computation cost, and run threat detection simulation in smart home testbed environments. At last, we summarize our FoCuS implementations.

6.2 Implementation Architecture

In this section, we illustrate implementation details regarding the IAM, BIdP, and dashboard modules respectively.

6.2.1 Identity and Access Control Management Module

As shown in Figure 3.5, the IAM module is composed of the identity hierarchy, BIdP driver, access control management, relationship management, and things management. As the critical module in FoCuS, the IAM module provides functions and APIs to manage identities, relationships, and access policies, through which subjects can create their hierarchical identity trees, register blockchain identities through the BIdP driver, establish relationships, and configure access control policies. We depict the UML class diagram of the IAM module in Figure 6.3, which includes 10 classes of realizing the previous mentioned five functionalities.

- Hierarchical Identity Management (HIdData Class)
- Relationship Management (RelationshipsData Class)
- BIdP Driver (BIdPData Class)
- Things Management (ThingsData Class)
- Access Policy Management (PoliciesData Class)

The IAMVerticle class is the main Vert.x verticle class that provides APIs to other FoCuS modules. In Table 6.1, we list all available APIs which could be divided into five categories(identities, things, policies, relationships, and bidp). For each category, there are two types of methods: read, write operations to the corresponding MongoDB JavaScript Object Notation (JSON) document. For instance, the first one (*/api/identities*) refers to a read method to list all identity information in the *hid_collection* MongoDB document while the second one (*/api/identities/upload*) refers to a write method to add new identities. Other modules call these APIs to manage identities and access policies.

6.2.2 Blockchain-based Identity Provider Module

The blockchain-based identity provider module comprises the service interface layer, transaction routing layer, and data storage layer, as shown in Figure 6.4. The service interface layer defines interfaces to the root identity operations, which include identity registration, update, revocation, and lookup. Subjects can regis-

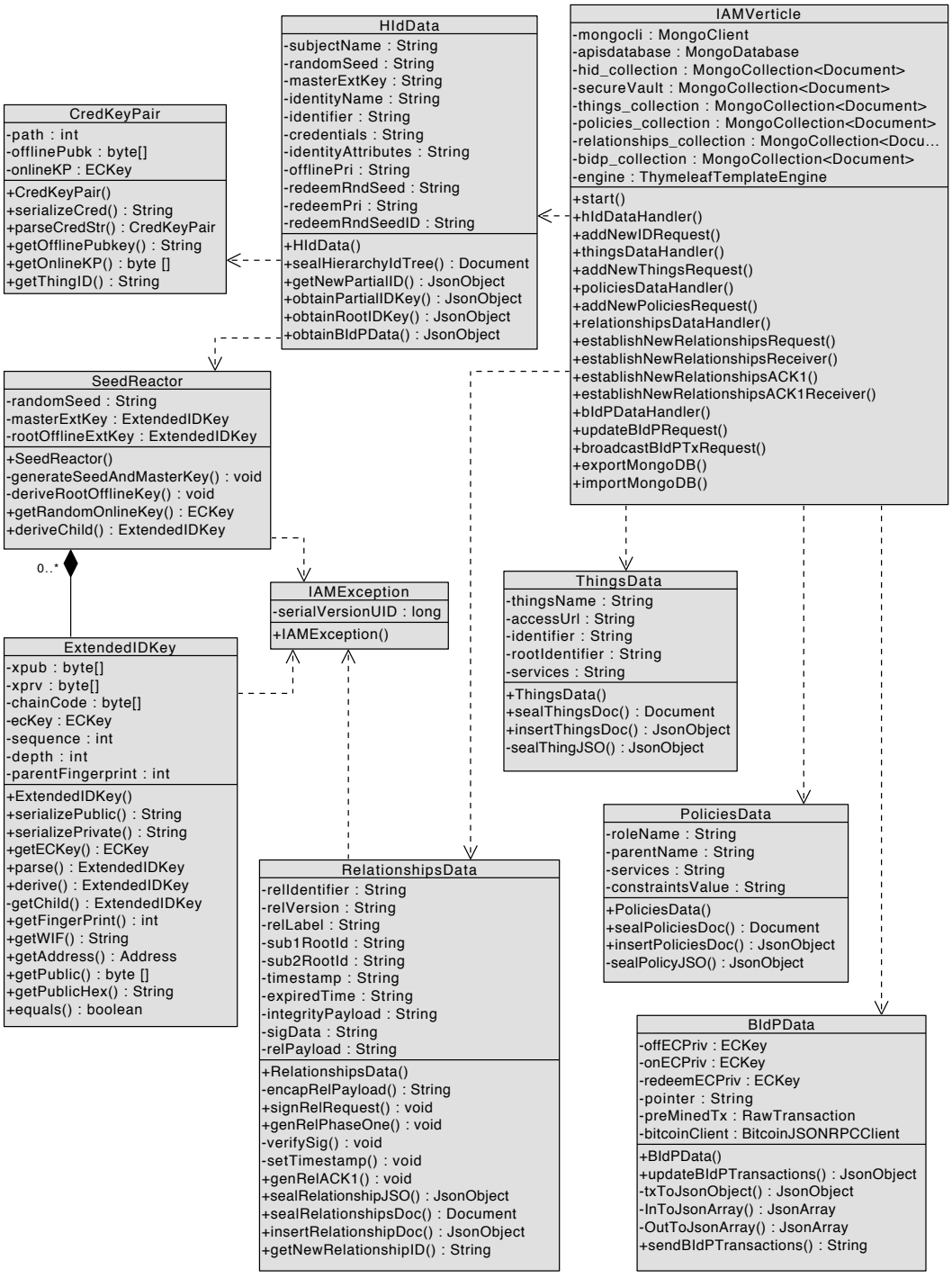


Figure 6.3 – The IAM module UML class diagram

Table 6.1 – The IAM module APIs list

APIs (/api)	Handler Functions
/identities	hIdDataHandler()
/identities/upload	addNewIDRequest()
/things	thingsDataHandler()
/things/upload	addNewThingsRequest()
/policies	policiesDataHandler()
/policies/upload	addNewPoliciesRequest()
/relationships	relationshipsDataHandler()
/relationships/establishrelreq	establishNewRelationshipsRequest()
/relationships/establishrelreceiver	establishNewRelationshipsReceiver()
/relationships/establishrelack1	establishNewRelationshipsACK1
/relationships/establishrelack1receiver	establishNewRelationshipsACK1Receiver()
/bidp	bIdPDataHandler()
/bidp/update	updateBIdPRequest()
/bidp/broadcasttx	broadcastBIdPTxRequest

ter, update, or revoke their identities using on-chain transaction interfaces. Alternatively, they can use the identity lookup interface to fetch identities when they receive service request messages to use them for authentication or authorization purposes. The transaction routing layer is responsible for receiving, interpreting, and dispatching transactions generated from the service interface layer. These transactions are firstly delivered to a daemon residing in computing nodes called transaction routing processor which filters different transactions. If the transactions are blockchain-write on-chain transactions, the processor will dispatch them to miners. If the transactions are blockchain-read off-chain transactions, the processor will execute the database operations after the identity authentication. The data storage layer comprises two types of data storage: the blockchain-based identity repository and the DHT based database. The former provides immutable identity records consensus which is formed by verifying on-chain transactions whereas the latter providing storage for non-identity information is a distributed database in P2P networks. After the generation of identity information by the identity and access control management module, subjects can make use of the necessary identity information to create the identity transactions. Then through interfaces in the service interface layer, subjects can deliver encapsulated identity transactions to the Bitcoin P2P network for mining immutable identity blocks. After the network achieving identity consensus, these blockchain identities could be used for authentication and authorization through identity lookup transactions.

When implementing the blockchain-based IdP module, we extended some signature functions in the Bitcoin stack (v0.16, Feb 2018). These extensions and modifications allow us to create and sign our identity transactions using $bitcoin - cli$ command tools, namely, *createrawtransaction* and *signrawtransaction*. Since new releases from the version 0.10 onward have enforced the transaction rule that only push operations are allowed in unlocking scripts, we made minor manipulations in the unlocking and redeem scripts as described by the Protocol BIdP update and revocation phases. Specifically, we placed the new public key new_pk to the redeem script. To this end, the subject needs to generate the replacing keys

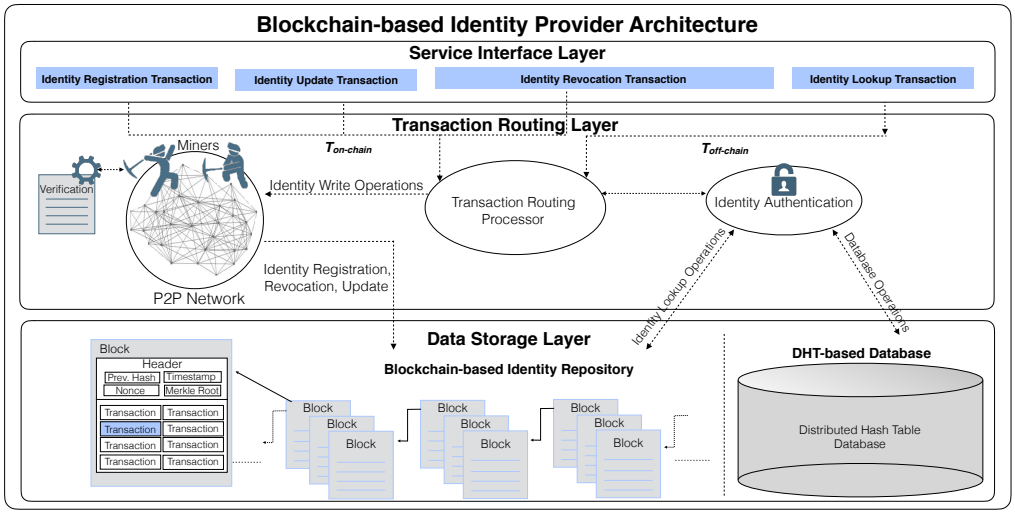


Figure 6.4 – Blockchain-based identity provider architecture

in advance. Thus, when trying to modify the keys, subjects have to construct and verify the following unlocking script:

$$0 < new\_\sigma_{sk} > < \sigma_{sk_o} > < \sigma_{sk_f} > 3 < new\_pk > \\ < pk_o > < pk_f > 3\ CHECKMULTISIG.$$

For experimentation purposes, we tested the BIdP module on the private Bitcoin *regtest* chain, the online Bitcoin *testnet* chain and even the *forked Bitcoin mainnet*. For instance, four types of identity transactions proposed in chapter 4, namely, pre_TX_{reg} [2], TX_{reg} [3], modifying pointer TX_{upd} [4], modifying key TX_{upd}/TX_{rvc} [5], can be found on the online Bitcoin testnet. Figure 6.5 shows the most sophisticated identity transaction: modifying key TX_{upd}/TX_{rvc} in Bitcoin testnet. As for the mainnet prototyping, we will present it in the dashboard GUI module.

6.2.3 Dashboard GUI Module

The dashboard module is a Web-based GUI and is designed to monitor and manage other modules. The dashboard module has two parts: a board for information regarding the IAM module and another board for information regarding the BIdP module. The IAM dashboard provides not only monitoring functionality over identity and policy information but also web interfaces through which we can manage identities and configure access policies. In the use case section, we present the dashboard GUI screenshots and demonstrate how to use these web interfaces to configure the IAM module. The BIdP dashboard, which aims at monitoring the status of the Bitcoin P2P network, locally resides in a BIdP node and can

[2] https://live.blockcypher.com/btc-testnet/tx/473986c4a3e28166f7751ca9d5a90f88b50a9b8055e58824bc3a61274e096b16

[3] https://live.blockcypher.com/btc-testnet/tx/4c25d8759cef46ce967fe48df0f13a266e1a15ae4f2139930584d1345009ab09

[4] https://live.blockcypher.com/btc-testnet/tx/8c2401aa3d00225720e48d5361dce77326284c5eff8264de044455c4083d8d46

[5] https://live.blockcypher.com/btc-testnet/tx/717832c79d7293f41ce6fe9b36367f567f6ad8a8c3fa28cc6686b41f0879f709

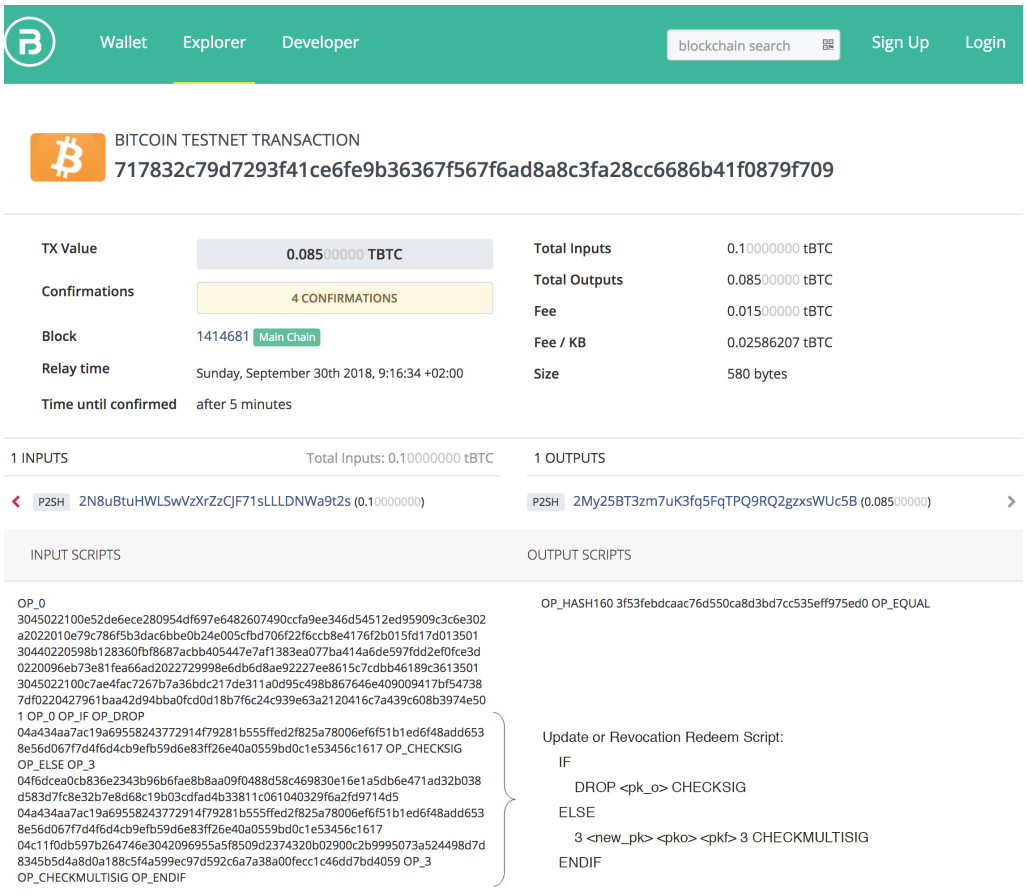


Figure 6.5 – Screenshot of an identity transaction mined in the Bitcoin Testnet

monitor other BIdP nodes connected to the local BIdP node in the Bitcoin P2P network.

6.3 Implementation Evaluation

In this section, we firstly illustrate our prototype with the case studies presented in chapter 3. Then, we analyze the performance of our FoCuS implementation from the perspective of scalability, interoperability, mobility, storage cost, transaction fee, and delay time in the real Bitcoin network, computation and communication cost, and lightweight verification overheads. At last, we set up a smart home testbed environment for threat detection module and run the simulation.

6.3.1 Case Studies Illustration

In order to illustrate the scenarios in the case studies, we build a Docker container image and deploy containers according to the requirements of the case studies. Specifically, we use Ubuntu (18.04) as the base image, install Java (11) and MongoDB (4.0), prepare dependencies environment for compiling our identity-featured Bitcoin stack, compile and install the modified Bitcoin client. When deploying the cluster containers, we build a P2P network of 4 nodes, representing four subjects in our scenarios. All deployed nodes are capable of running miners. In our

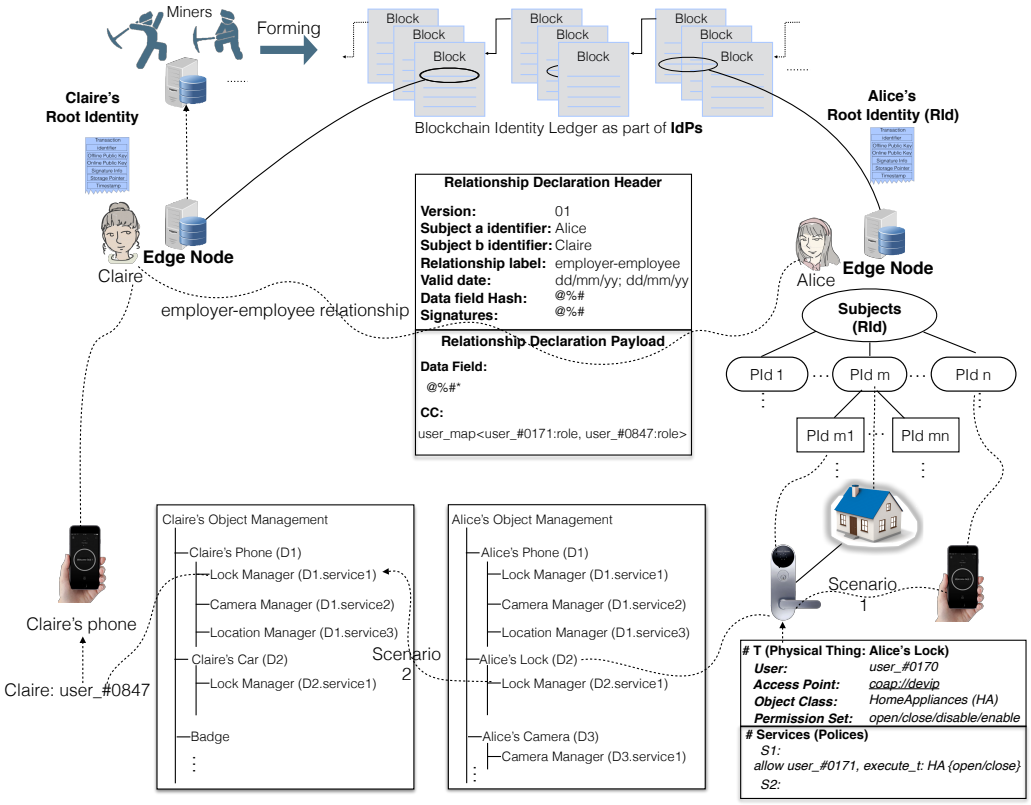


Figure 6.6 – Scenario 1 and scenario 2 overview

demonstration, each node is running and can be accessed through its dashboard using local IP address and a port number (i.e., $3500x$, x is the node number).

- Node-1: Alice – Dashboard GUI (http://localhost:35001)
- Node-2: Claire – Dashboard GUI (http://localhost:35002)
- Node-3: The Hospital – Dashboard GUI (http://localhost:35003)
- Node-4: Bob – Dashboard GUI (http://localhost:35004)

6.3.1.1 Case 1 and Case 2 GUI Representations

Case 1 and Case 2 involve two subjects: Alice and Claire. Figure 6.6 depicts Scenario 1, where Alice can open her house door using her cellphone, and Scenario 2, where Alice allows Claire to open her house door using Claire's cellphone within a certain period of time. Alice, as an individual subject, uses our FoCuS prototype to securely manage her things, and hence she can create a hierarchical identity tree to manage her things listed in Figure 3.8. With the increment of application scenarios, Alice could dynamically add more partial identities for her IoT devices or objects. For instance, Alice uses the Partial Identity m (PId m) to identify her house and the Partial Identity mn (PId mn) to represent the door lock in her house. Following the object management, the door lock needs to be registered under our framework to become the identifiable and operable thing. Specifically, the door lock should be configured via assigning the administrative user, access point, object class, and corresponding permission set. Then, Alice, as the owner, could take advantage of native services provided by the door lock to form (composable)

services which could be exposed to external users. In order to interact with others, Alice still needs to post her root identity transaction to the Bitcoin P2P network, which is composed of edge nodes in our demonstration, to reach the consensus on her identity. Once miners mine the identity into a block, and Alice gets enough confirmations from the network, her identity is admitted by the Bitcoin P2P network. Upon the successful identity blockchain registration, subjects can take advantage of registered blockchain identities to establish relationships and grant access permissions.

Figure 6.7 illustrates the sequence diagram of these two scenarios. Alice firstly generates her root identity of the hierarchical identity tree (Step 1) and then post the registration transaction to blockchain based identity provider to reach identity consensus (Step 2). Step 3-4 refer to the object's registration process, after which the owner can deploy services, configure access policies, and even dispose of the object. Step 5 demonstrates the authentication process of Scenario 1, in which the owner tries to access their own devices using the identity of the corresponding device. In other words, Alice uses the private key of the lock to unlock the lock. When the private key is kept in her cellphone, and it becomes Scenario 1 as shown in Figure 6.6. Scenario 2 describes that Alice wants to hire a housekeeping – Claire to clean her house regularly, where Claire is allowed to open the lock without bothering Alice. Therefore, Alice and Claire need to establish a relationship firstly (Step 6). The establishment of a trust relationship between two subjects requires to be negotiated by the two parties. Upon the negotiation, the relationship is confirmed, and the corresponding privileges are granted (Step 7-9). Figure 6.6 also shows the established relationship, labeled as employer-employee relation, and indicates that access permissions are granted to Claire based on the contentment of Alice.

Creating Identity Tree

Alice uses a Web browser to access the node-1 client (http://localhost:35001). In the identity hierarchy Web page, Alice fills in basic identity information (namely, *Identity Name* and *Identity Attributes*) to create her identity tree using the GUI of creating an identity tree as illustrated in Figure 6.8. According to the case study proposed in Figure 3.8, we present the complete Alice's identity tree in Figure 6.9. All identities are organized using the identity management tree, in which the root node stands for Alice root identity, and other nodes are created for sub-domains or things such as a home, car, and wearable devices.

Adding Things

In the things Web page (see Figure 6.10), Alice adds all her owned things with their required information (namely, *Things Name, Things AccessUrl, Things Domain*, and *Things Services*) to register things information with the IAM module. When adding a new device to the system, Alice also can register new devices by calling the provided REST APIs using the devices. When registering a thing to the IAM module, a new partial identity is created for the thing whose parent domain is designated through the *Things Domain* parameter. All services provided by the new thing are also registered to the IAM module so as to configure access policies. Figure C.1 shows Alice's things registry view.

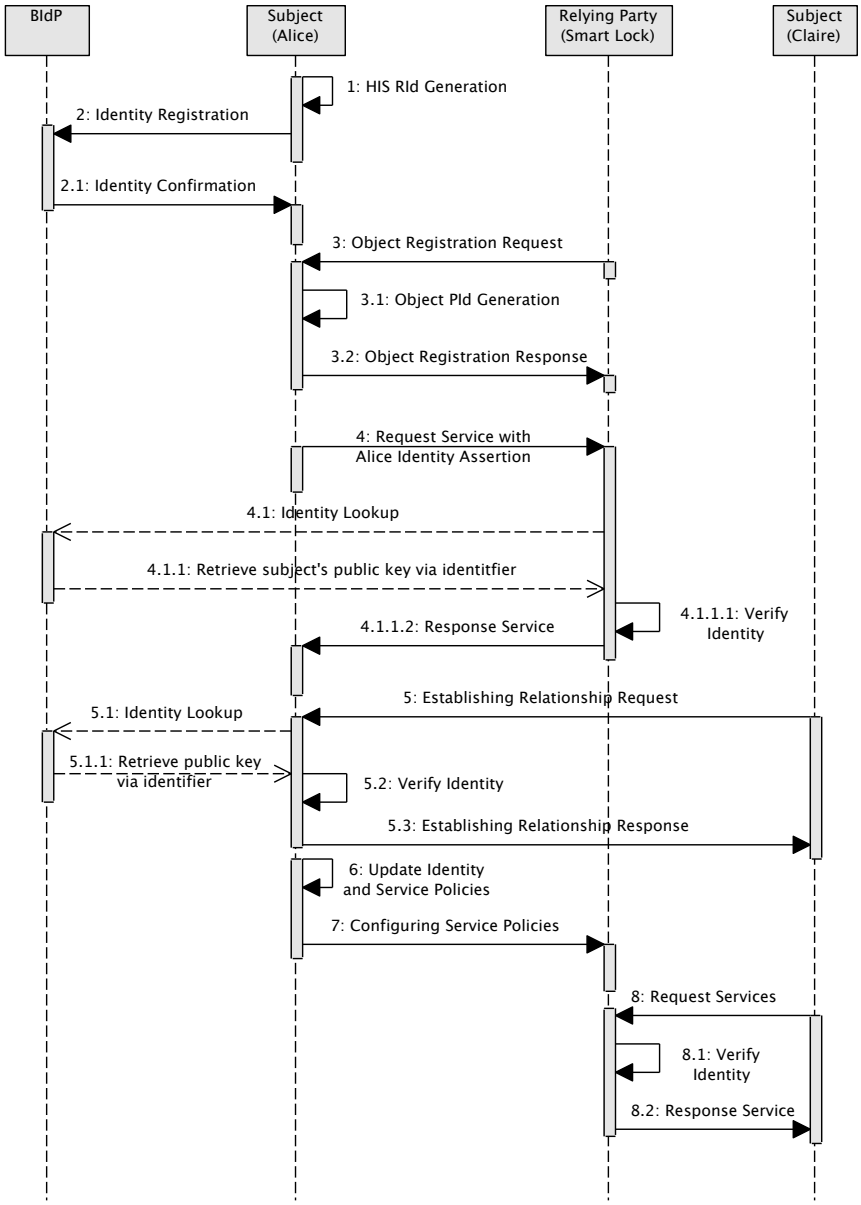


Figure 6.7 – Scenario 1-2 sequence diagram

Blockchain Identity Provider Transactions

After generating the root identity, Alice can use the blockchain based identity provider driver (BIdP) in the IAM module to register her identity to the blockchain based identity provider. Since the registration of blockchain identities requires Alice to provide bitcoin balance, we predefine a mining address (*1Ez1ZNPLPb8rotec EBZWtNWNx8oijzwxQv*) for the sake of the demonstration purpose, which has enough bitcoins after enabling miners in the network. When updating blockchain identity transactions, we can see the available transactions that need to be delievered to the Bitcoin P2P network as shown in Figure C.2. Alice can broadcast these transactions to the network one by one. The dashboard of the modified Bitcoin network is shown in Figure C.3, through which we can monitor the status

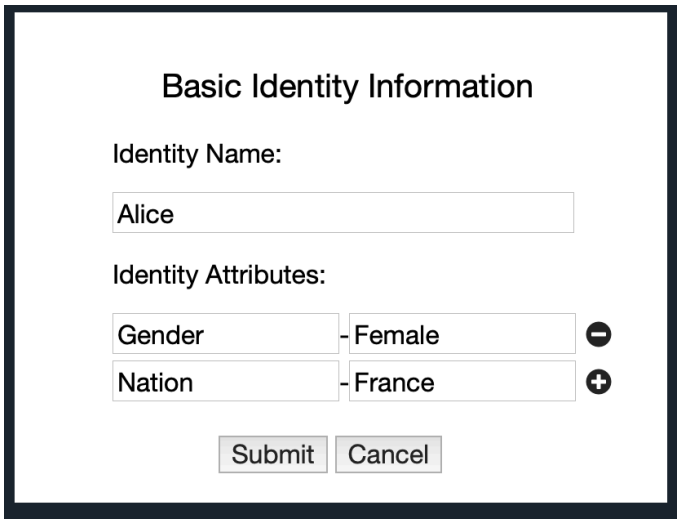


Figure 6.8 – Creating an identity

of the private Bitcoin network such as mined blocks, transactions in the memory pool, sealed transactions, and connected peers.

Adding Policies by Creating Roles

Alice can create an access policy proposed in Scenario 2 (Alice allows Claire to open her house door using Claire's cellphone within a certain period of time) through adding a role for her house door as shown in Figure 6.11. Alice needs to give the role name, granted services, and access constraints such as time period (e.g., 1-3 p.m). All created policies are listed in the access policy Web page. From Figure C.4, we can see, Alice added one policy that allows Claire to open the main door during 1 p.m. - 3 p.m.

Establishing Relationships and Granting Permissions

When Alice decides to hire Claire to clean her house regularly, Claire should search for the unlock service provided by Alice's main door. As illustrated in Figure 6.12, Claire has to fill in the required information in the relationship (namely, *Relationship Label, Alice's Root Identifier, Service Provider Access Point,* and *Mapping Users Information*). After filling in the access point, Claire clicks the search button to automatically search Alice's list of services and then chooses the door unlocking service provided by Alice's main door. The *Requester Capability* means which identity Claire wants to use to access the main door service as provided by Alice. After Claire submitting the establishing relationship request, Alice can find the requested unconfirmed pending relationship in the relationship Web page (see, Figure C.5). Once Alice validates the relationship request, she designates a relationship expiration date (see, Figure 6.13). Then, Claire can get a response regarding the confirmed relationship in Claire's relationship Web page as shown in Figure C.6.

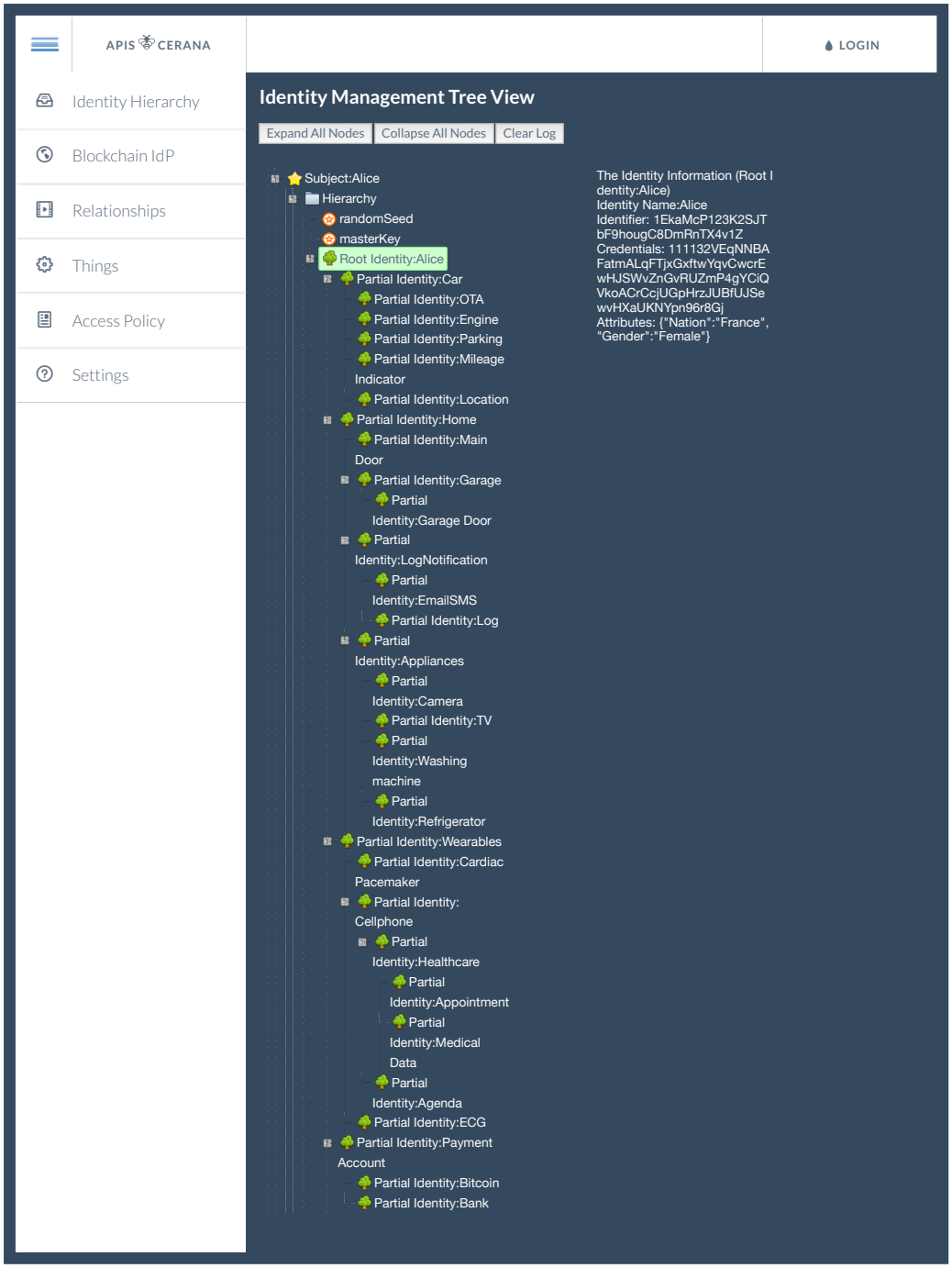


Figure 6.9 – Alice's identity tree

Figure 6.10 – Adding a thing

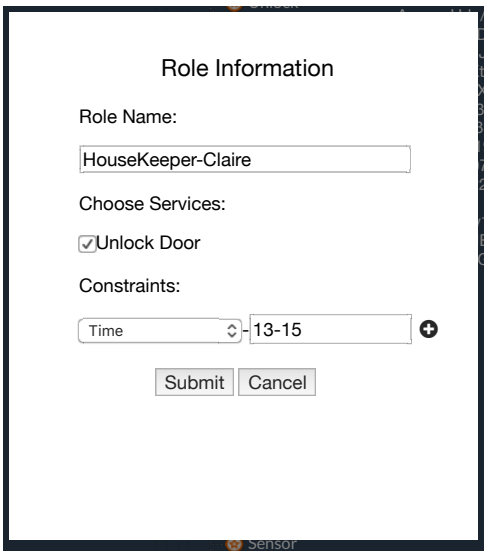


Figure 6.11 – Adding a role for things

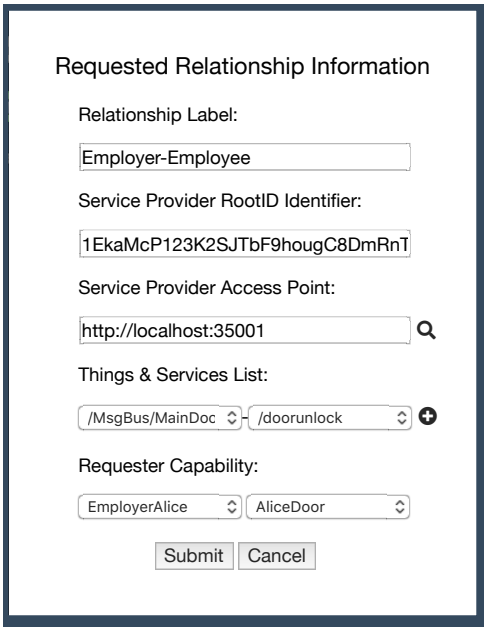


Figure 6.12 – Relationship request

Figure 6.13 – Relationship expired date

6.3.1.2 Case 3 and Case 4 GUI Representations

The hierarchical identity tree creation of the hospital is similar to the individual hierarchical identity tree creation process for Alice (see, Figure 6.8). Besides, the collective still needs to set access permissions inside the collective in that collectives can be not only the manager of things like an individual but the container of individuals. Therefore, collectives have to assign access permissions to partial identities that represent other individuals. The process of granting access permissions is identical with the process described in Alice granting permissions. Firstly, the hospital needs to adding policies by creating new roles. Then, the hospital can establish relationships and granting permissions. The only difference is establishing relationships with itself. In fact, there is no difference between individuals

and collectives, which can transform each other. Scenario 3 and Scenario 4 can be considered as the same scenario.

When Dr.Bob hired by the hospital, the hospital creates a partial identity, which will represent Dr.Bob in the hospital. The hospital can configure access policies to the partial identity. If Dr.Bob wants to use the partial identity in the hospital (namely, Scenario 5), Bob needs to establish the relationship with the hospital which is similar to the establishing relationship process between Alice and Claire in the previous section. Similarly, scenario 6 describes that Alice and the hospital need to establish patient-hospital $S2S$ relationship and grant access permissions (namely, taking an appointment service). After making the appointment, the hospital can assign a doctor to Alice (namely, Scenario 7) which is the granting permission inside collectives like Scenario 3 or Scenario 4.

6.3.2 Performance Analysis

Performance evaluation is essential to make sure that our contributions scale up for the IoT-based systems and services. From previous chapter 4 and chapter 5, we can see that our IoT secure infrastructure – FoCuS is built from two frameworks: the blockchain based identity management framework (BIMSIT) and the decentralized IoT access control framework (DITAC). Hence, the performance should be evaluated from these two frameworks separately.

6.3.2.1 Characteristics Analysis

Traditional identity management systems evaluation relies on the performance of their underlying servers, which may not properly scale up in distributed IoT environments composed of billions or even trillions smart objects and devices. The blockchain technology gives us the ability to design the scalable distributed system in trustless environments. However, the performance of blockchain based systems still needs measurement to quantify and compare results. The blockchain performance is still in its infancy and recent researchers have proposed analysis frameworks for the blockchain systems. Gervais et al.[Ger+16] analyzed proof-of-work based blockchain systems like Bitcoin, Litecoin, Ethereum with different operational parameters. Dinh et al.[Din+17] proposed their evaluation framework for private blockchain systems, in which they analyze the blockchain systems from the consensus, data model, execution layer and application layer. They also quantify the system from specific metrics including throughput, latency, scalability, and fault tolerance. Evaluating the performance of the blockchain system, in essence, refers to evaluating the scalability of distributed systems, which needs a holistic consideration about the cost and Quality of Service (QoS). We could follow the proposed framework of evaluating the distributed systems in [JW00] to designate some parameters to the predefined general family of metrics, which include the rate of providing services (λ), the quality of services (QoS) and the cost of providing services (C) in the Equation 6.1 so as to build the evaluating framework for blockchain identity management systems.

$$\psi = \frac{F\left(\lambda_2, QoS_2, C_2\right)}{F\left(\lambda_1, QoS, C_1\right)} \tag{6.1}$$

We analyze the scalability of our BIMSIT framework using the scalability evaluation theory for heterogeneous and distributed systems and conclude that the BIMSIT is scalable in that the mean delay time T does not increase as the number of identity provider nodes increases. On the contrary, the more nodes with full identity repository, the smaller the mean delay becomes. The proof can be found in Appendix A. Since the DITAC framework is decentralized and resident in secure gateways or Fog nodes to grant access permissions to others, the increasing number of users or nodes in the network has nothing to do with the single user experience of our decentralized access control framework. It only depends on the time of establishing relationships between two subjects and the communication cost between devices which are not determined by the scale of the network. Therefore, the scalability of the DITAC will not be the bottleneck of scalable FoCuS.

Mobility is another capability of our BIMSIT. Benefiting from the SPV (Simplified Payment Verification) [Nak08] in Bitcoin, our BIMSIT could verify identity transactions without a full node, which makes our BIMSIT could be installed on mobile IoT devices with limited resources. Similarly, the decentralized DITAC enables access control policies in Fog/Edge nodes or even end nodes like cellphones which are mobility-native devices. There is no federation concept: each individual or collective could join the open source BIMSIT without the bothering of across-domain problems, which increase interoperability of different entities due to the universal identifier naming standard. Moreover, built-in three types of relationships from the DITAC framework weave a massive and extensible social network that unifies all IoT entities including human, organizations, government, things, applications, services.

Blockchain matches identity management characteristics: in identity management systems, operations (i.e., registration, revocation, and update) are called less than lookup operation: fast to read and verify (LDAP-based Identity Directory), slow to register and update (Identity Provider), which is identical with our blockchain-based IdP operations. The shifting from traditional IdPs to distributed blockchain-based IdPs not only eliminates the single point failure but provides all subjects with an always online IdP service, which significantly simplifies the interactive communication as shown in Figure 1.1 and boost the performance of the BIMSIT. Every node could be IdPs as long as this node runs full identity blockchain. Thus, ubiquitous IdPs shorten or even eliminate the time-consuming IdP discovery. As for the online DHT, it is only used for secured backup usage unless the subject wants to use the identity from new devices or loses the identity set for some reason. Therefore the performance of the DHT-based database will not be a bottleneck for our system.

6.3.2.2 Evaluation Metrics

Besides characteristics analysis, we also give some performance metrics based on our prototyping in this section, which cover storage cost, transaction fee, and delay time in the real Bitcoin network, computation and communication cost, and lightweight verification overheads in terms of CPU, RAM, and bandwidth.

On chain Storage Cost. The BIMSIT framework is a scalable identity solution in terms of storage cost. Firstly, the hierarchical identity tree separates the root identity and partial identities. Only root identities, standing for individuals or collectives, are on-chain identities, which reduces the blockchain bloating. Secondly, only identity registration, revocation and update operations with low usage frequency can generate the on-chain transactions, which also extremely alleviates the bloating problem in the blockchain. Thirdly, according to our identity solution prototyping, we can see the size of each type of transaction: pre_TX_{reg} (222 bytes), TX_{reg} (269 bytes), modifying pointer TX_{upd} (455 bytes), modifying keys TX_{upd}/TX_{rvc} (580 bytes). Therefore, for each on-chain transaction operation, the maximum storage cost is only 580 bytes. Besides, unlike the Bitcoin system, the increasing number of Bitcoin users will dramatically inflate the Bitcoin blockchain in that each new user will interact with not only one user. In contrast, identity operations in the BIMSIT framework, only allow the owners of identities to update their transactions. As for the DITAC framework, it is not necessary to consider the storage cost in that the proposed access control interactions exist in users' personal devices.

Bitcoin Network Fee and Delay Time Evaluation. According to the on-chain storage cost of our identity transactions, we use the data of last year from 22 March 2018 to 23 February 2019 as references to evaluate the fee and delay time of identity on-chain transactions in the real Bitcoin network. Specifically, we looked up the average Bitcoin transaction fee (that is, 25 satoshi/byte) and average Bitcoin transaction confirmation time in minutes (that is, 13.44 minutes) from the Bitcoin network monitoring statistics website[6]. Also, we take the largest transaction (580 bytes) as an example to draw Figure 6.14. We can see that the total fee of identity transactions is proportional to the price the user is willing to pay. When the price is above 25 satoshi/byte, the total fee is 14500 satoshi, and the transaction will be sealed into the next block in 13.44 minutes. Consequently, the largest identity transaction (588 bytes) will cost \$0.58 (current bitcoin price: \$4000). However, following our assumption in Appendix A, the on-chain operations are relatively rare compared with off-chain lookup operations. Unless losing root private key, users do not have to do on-chain operations. Besides, we have to notice, the fee and delay time are closely related to the network. If we setup up a private chain network, adjust the interval of blocks and increase the block size, results of delay time and fees can be changed. For instance, if we adapt BIMSIT to litecoin network, the average delay time will reduce to a quarter of Bitcoin delay time.

Computation and Communication Cost in Authentication. In order to estimate the computation and communication time cost of the authentication process in the BIMSIT framework, we define T_{sig} and T_{ver} as computation time cost of signing data and signature verification while T_{A-B} as communication cost between two parties A and B. Since authentication is the basis of other security services like authorization or access control, we compare our authentication scheme with OpenID [RR06] – the most popular decentralized identity management solution. As shown in Figure 4.4, the authentication only needs

[6]https://bitcoinfees.info, access March 2019

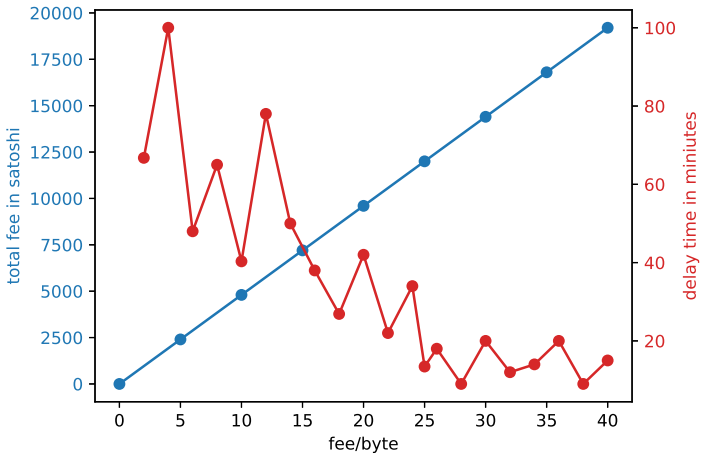


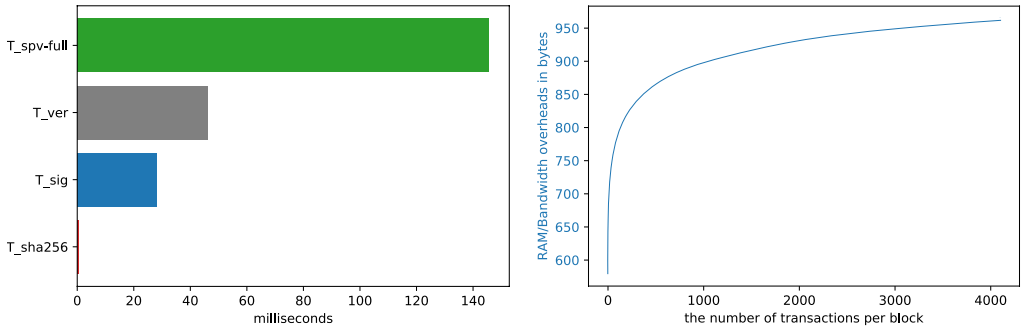
Figure 6.14 – Bitcoin network cost fees and delay evaluation

Table 6.2 – Computational and communication cost comparison

	Computation Cost	Communication Cost
BIMSIT Authentication	$T_{sig} + T_{ver}$	$2 * T_{SUB-RP}$
OpenID (Implict model)	$T_{sig} + T_{ver}$	$5 * T_{SUB-RP} + 3 * T_{SUB-IdP} + T_{IdP-RP}$
OpenID (AC model)	$T_{sig} + T_{ver}$	$5 * T_{SUB-RP} + 3 * T_{SUB-IdP} + 3 * T_{IdP-RP}$

four steps: generating Identity Assertion (T_{sig}), authentication request communication (T_{SUB-RP}), verify IA (T_{ver}), authentication response communication (T_{RP-SUB}). However, according to the OpenID specification, we describe the authentication process and calculate the computational and communication cost of OpenID implicit model: 1) a subject sends the service request to the relying party using the identity issued by third-party IdP (T_{SUB-RP}); 2) the device initiates IdP discovery procedure and then redirects the user to the corresponding IdP for authentication ($T_{RP-SUB} + T_{SUB-IdP}$); 3) the user provides the password to IdP for authentication ($T_{IdP-SUB} + T_{SUB-IdP}$); 4) IdP verifies the password and redirects the user to the device with ID Token which is a signed request and could be verified to identify the user's identity ($T_{sig} + T_{IdP-PR} + T_{RP-SUB} + T_{SUB-RP}$); 5) After validating the ID Token, the device could provide services to the user ($T_{ver} + T_{RP-SUB}$). Since T_{A-B} equals T_{B-A}, we could have Table 6.2, in which we also compare the cost of OpenID Authorization Code model with additional operations using the authorization code to exchange ID token. In Table 6.2, we did not take into account the Identity Provider discovery in the federated identity management which could be a bottleneck to the IoT low-latency requirement. However, in our solution, identities stored in blockchain are admitted by the entire P2P network. Therefore, the distributed identity provider eliminates the IdP discovery process and remarkably saves the communication cost of the IdP discovery process.

Lightweight Verification Node Overheads. Mobility is another capability of our BIMSIT. Benefiting from the SPV (Simplified Payment Verification) [Nak08] in Bitcoin, our BIMSIT could verify identity transactions without a full node, which makes our BIMSIT could be installed on mobile IoT devices with limited resources. In order to testify this, we launch experiments of the lightweight



(a) CPU and communication overheads (b) RAM/Bandwidth overheads

Figure 6.15 – Lightweight verification node overheads

SPV nodes on Raspberry Pi3 B (with 1.2GHz, 64-bit, quad-core, ARMv8 CPU and 1GB RAM). Besides the computation and communication cost of BIMSIT authentication summarized in Table 6.2, SPV nodes have to inquiry full nodes to get the identity transaction and a set of hash values used for authentication. Therefore, the hash calculation time (T_{sha256}) should be added to computational cost ($T_{sig} + T_{ver} + kT_{sha256}$), where k is the number of hash values of the merkle authentication path [Ant14]. Considering the Bitcoin block size is about 1MB (4096 transactions/block), we have $k \in [1, 12]$. Also, compared to the signature operations (sign and verify), the hash time (kT_{sha256}) is negligible. On Raspberry Pi, we run the authentication program 50000 times and get the average CPU overheads ($T_{sha256} \approx 0.5ms$, $T_{sig} \approx 28ms$, $T_{ver} \approx 46ms$) as shown in Figure 6.15 (a). Moreover, we also estimate the extra communication cost ($T_{SPV-FULL} \approx 145ms$) with full nodes (Bitcoin testnet) in Figure 6.15 (a), which should be added to previous communication time ($2*T_{SUB-RP} + 2*T_{SPV-FULL}$). From the comparison, we can see, the communication cost is much larger than the others. Finally, according to the number of hash values (32 bytes) in authentication path from SPV nodes, we can conclude that the memory and bandwidth are logarithmically related to the number of transactions in one block as drawn in Figure 6.15 (b), in which the maximum consumption is 964 bytes: maximum identity transaction (580 bytes) adds the maximum authentication path (k * 32 bytes, $k = 12$).

6.3.3 Threat Detection Experimental Results

In order to test our threat detection module, we build a smart home testbed (see Figure 6.16), which has all the characteristics and functionalities of actual smart homes such as sensors, actuators, automation systems, and communication channels. In our testbed, the user can monitor the variables and control elements using a variety of protocols (e.g., Wi-Fi). Variables include temperature, distance, motion, current, humidity, and illumination. The elements to control (actuators) are LED lights, lamps, ventilators, door lock, and electric sockets (where televisions can be connected). The information from the sensors is acquired by an Arduino board [Bel14] every millisecond but updated in memory every 5 milliseconds. The main tasks of the Arduino board are: 1) collect information from sensors, 2) analyze device usages looking for abnormalities such as high-frequency command

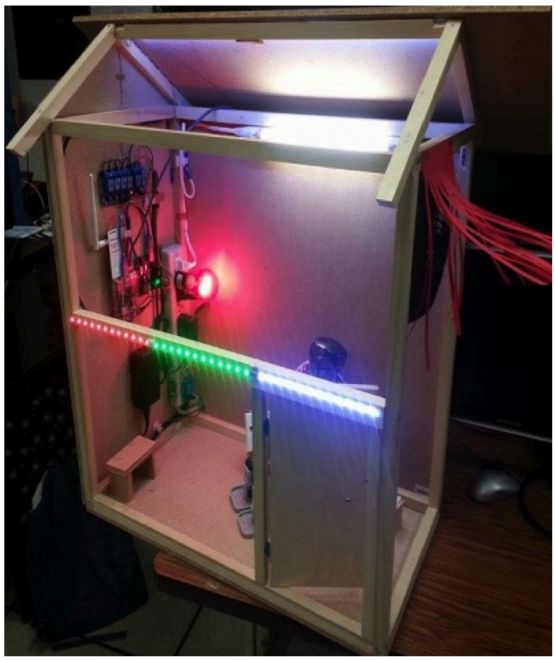


Figure 6.16 – Smart home testbed

issues or excessive power consumption, 3) trigger alerts in case an abnormality is detected, and 4) send the collected information to the secure gateway (including the alerts) through serial communication. The monitor and control tasks can be performed locally by accessing our secure gateway, and remotely by using cloud services. The secure gateway (edge node) is built in a Raspberry PI system [Bel14] that runs under Debian (Linux). The main tasks of the Raspberry are 1) receive the information from the Arduino board, 2) publish the information in a website hosted in the same Raspberry, 3) enable Wi-Fi communication for local control/-monitor of the testbed, and 4) send all the information to the cloud. A noticeable characteristic of the secure gateway is the ability to perform self-protection by using Anomaly Behavior Analysis as we did in [PH16].

Our preliminary experimental results for intrusion detections show that our framework can effectively detect both known and unknown threats with high detection rates and low false positive alarms in cyber-physical devices.

6.3.3.1 Threat Detection Experimental Setup

Connected devices. We used two DC motors as end-nodes according to the proposed decentralized scheme in Figure 5.6. For DC motors plant, the model represented in state space is given by Equation 6.2 and Equation 6.3.

$$\begin{bmatrix} \dot{x}_{i1} \\ \dot{x}_{i2} \end{bmatrix} = \begin{bmatrix} -\frac{b_i}{J_i} & \frac{k_{im}}{J_i} \\ -\frac{k_{ib}}{L_i} & -\frac{R_i}{L_i} \end{bmatrix} \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L_i} \end{bmatrix} U \tag{6.2}$$

$$y_i = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}, for(i = 1, 2) \tag{6.3}$$

where $\dot{x}_{i1}$ is the angular speed of motor i and $\dot{x}_{i1}$ is the current for motor i. The actual parameters for these motors are shown in Table 6.3.

Adaptive model. A neural network configured as a nonlinear autoregressive with external input (NARX) architecture is designed as the adaptive model block [Bil13]. The neural model fits the input-output map of DC motors. The representation of the neural model is

$$\begin{aligned} y(t) = f[&y(t-1), y(t-2), ..., y(t-n_a), \\ &u(t), u(t-1), u(t-2), ...u(t-n_b)] + \in \end{aligned} \tag{6.4}$$

where n_a, n_b are the order of the time delayed of outputs and inputs respectively; $\in$ is the error term and f is the feed-forward neural network.

Controller. It is well-known that the majority of industrial applications rely on Proportional- Integral-Derivative (PID) controllers [Zhu09]. For that reason, we chose a PID controller, which will be tuned to control the angular velocity of DC motors.

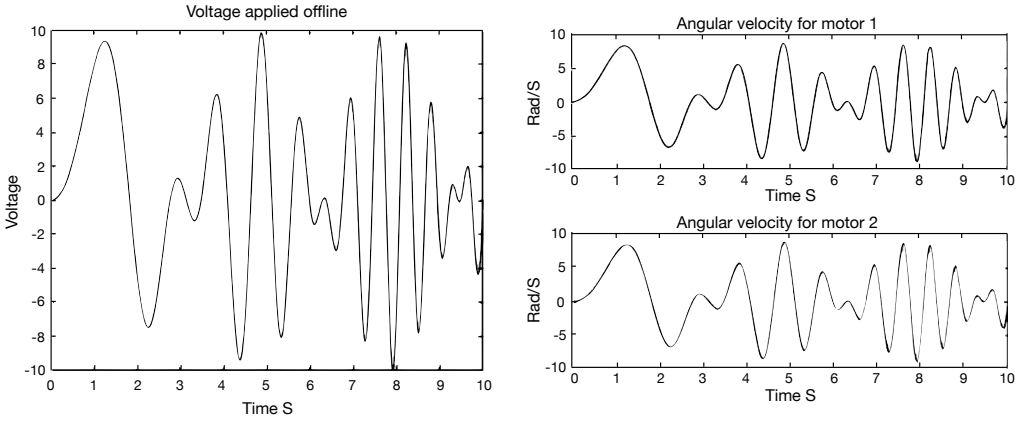
Updating learning law. The neural network is trained with Back Propagation learning rule based on the Levenberg Marquardt algorithm [Hay+09], which is specifically designed to minimize sum-of-square error functions

6.3.3.2 Threat Detection Experimental Results

DC motors are excited offline using a chirp signal modulated by a sinus component with amplitude 10 units and a linear variation from 0.1 to 2 Hz. This stage is necessary to excite all possible frequencies (modes) of DC motors and to get all its range of operation. The modulated chirp signal and respective output response for both motors are shown in Figure 6.17. Input-output data is used to train the neural network, which is designed with a single hiding layer with 5 neurons;

Table 6.3 – Actual parameters for two nodes

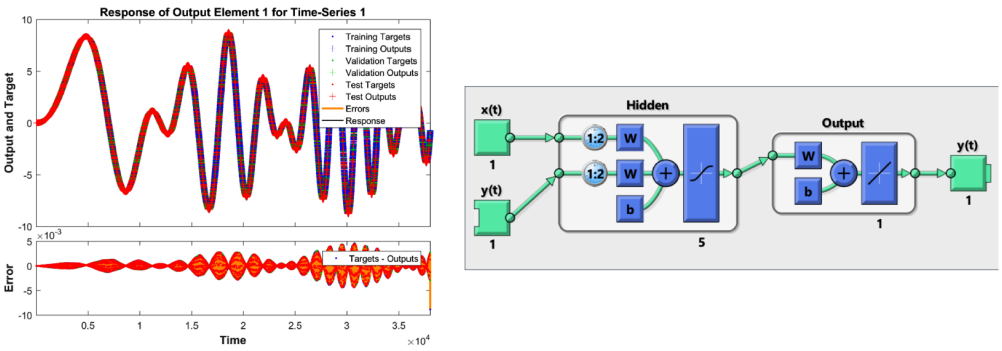
Node	Type of node	Parameters
1	DC Motor	$R_1 = 6.65$; $L_1 = 0.0016$; $J_1 = 0.001969$; $k_{1b} = 0.920608$; $b_1 = 0.0281$; $k_{1m} = 0.920608$; $V_1 = 10$
2	DC Motor	$R_2 = 10.6$; $L_2 = 0.00082$; $J_2 = 0.00000116$; $k_{2b} = 0.0502$; $b_2 = 0.005$; $k_{2m} = 0.0502$; $V_2 = 10$



(a) Voltage applied to DC motors to get input-output data

(b) Output response of motors

Figure 6.17 – Experimental motor data



(a) Neural network performance

(b) Neural network architecture

Figure 6.18 – Neural network experiment

two delayed input-outputs are selected. Data are divided as follows: 70% is used to train the neural network; 15% is used to validate, and the remaining 15% is selected to test the identification performance of the neural model. Figure 6.18 (a) shows the training process of the neural network, whose architecture is displayed in Figure 6.18(b). The learning block generates synaptic weights. The neural weights obtained by the learning law are listed in Table 6.4 and Table 6.5.

Table 6.4 – Synaptic weights

Hidden Layer Weights			
Weights{1,1}		Weights{1,2}	
-0,1532271907	-1,0154084414	-0,4542751972	1,5057835420
-0,8451821460	0,9395536793	0,0387100069	0,4292194356
-0,4002233848	-1,4983336550	1,8145677946	-0,1946276927
0,008593647136	0,01614434075	0,6221740272	-0,3135355548
-1,041471726	1,0004194457	-0,3160268995	-0,1904878580

Notice that Table 6.4 and Table 6.5 represent the normal or nominal behavior of motors, modeled by an adaptive neural model, such model can be used by the upper layers in our framework to identify the nominal operation of nodes. Also, it is worth to mention that end nodes are modeled by an adaptive model, no mathematical description is required. Let us call this data as Data nominal. To testing out our approach, two operational conditions are simulated.

Table 6.5 – Neural weight results

Output Layer Weights		
0,7164304868	0,4537938692	0,031948627
2,667628341	-1,1926025389	
Hidden Layer bias		
2,3050228090	0,2405151853	-0,4694639821
-0,6588869711	1,2699794119	
Output Layer Bias		
-0,2728024716		

Operational Condition 1 The motors operate in closed loop mode with PID controllers tuned to track a constant velocity with reference signals shown in

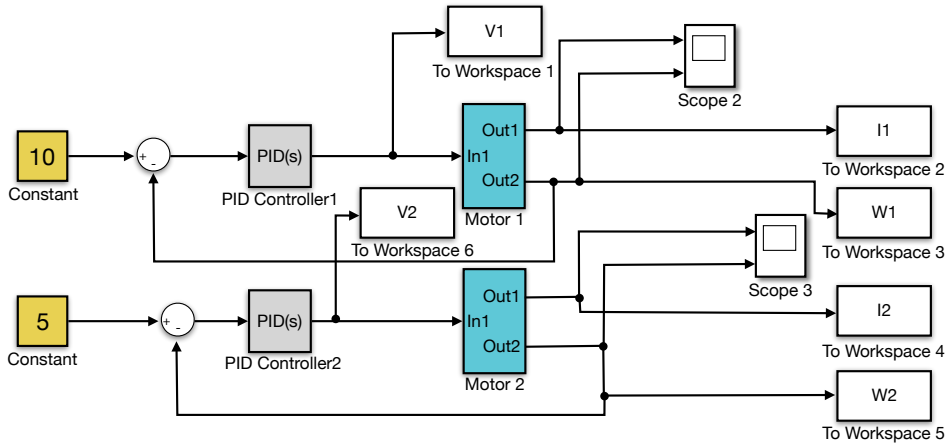


Figure 6.19 – PID controllers tracking a constant velocity command

Figure 6.19, where references are 10 rad/s for motor 1 and 5 rad/s for motor 2 and reference signal, controller and motors are highlighted. These conditions are very common in industrial applications. The synaptic Weights and Bias under these conditions are obtained and labeled as *Data_actual*. For space restriction, this data is shown in Table 6.6 just for the hidden layer.

Table 6.6 – Neural weights for operational condition1. Only hidden layer is shown

Hidden Layer Weights	
Weights{1,1}	Weights{1,2}
-1.2531	0.7435
0.2112	-1.6244
0.5725	-0.7662
-0.226	0.153
-1.1082	1.2085

Operational Condition 2 An abnormal operation is simulated where motors are suddenly out of velocity, i.e., the motor goes from constant speed to zero instantaneously. This fault condition can be considered as severe due to industrial processes normally operates under speed profiles that never change its operation from constant speed to zero in a short time span. Under condition two, the local controller for each node can recover to a healthy condition. If fault operation persists the upper layers in the frame work applies the appropriated correction to restore the nodes to their nominal actual operations.

Table 6.7 – Abnormal operation example

Hidden Layer Weights	
1.1171	-1.1133
-1.2926	1.2074
-1.1405	1.1242
-0.2295	-0.9482
-0.0739	1.5081

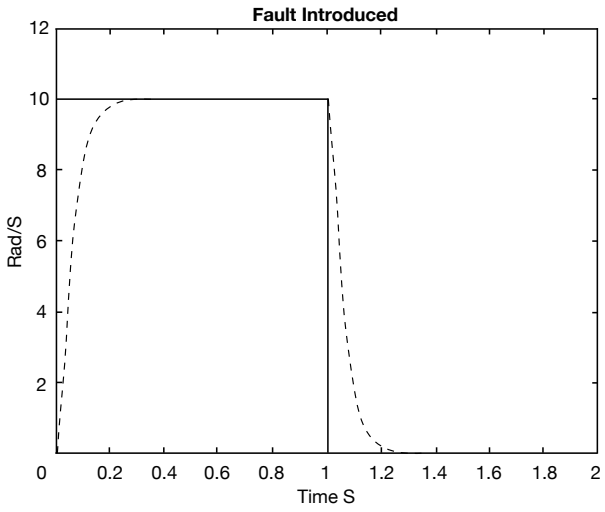


Figure 6.20 – A velocity fault simulated at 1 sec of actual operation. Motor goes from a normal speed of 10 Rad/S to zero

The input signal and the corresponding output of motor 1 is depicted in Figure 6.20. Even though in the fault operation case, the controller can follow the signal input, an action is required by the upper levels in order to assess if this operation is occasioned by a dysfunctional operation of nodes or by an intrusion external command, in any case, the neural network data obtained in this operative mode is label as $Data\_fault$. Equation 6.5 is applied as a criterion to distinguish.

$$\mathcal{K} = \mathcal{L}_2|Data_{actual} - Data_{fault}| = \begin{bmatrix} 5.6178 & 3.4477 \\ 2.2614 & 8.0191 \\ 2.9344 & 3.5736 \\ 0.0000 & 1.2126 \\ 1.0698 & 0.0898 \end{bmatrix} \tag{6.5}$$

It is clear the mismatch between the matrix $\mathcal{K}$ and the $Data\_nominal$ reference matrix shown above. If we bound the entries of $\mathcal{K}$, it is possible that the risk management and the action handling units can take the appropriated commands to restore the CPS to a healthy condition. Even more, due to that PID controllers remains the transient error close to zero and, the adaptive neural model converges to the dynamics of the plant fast enough, it is possible to establish according to Equation 5.9 and Equation 5.10 that this fault condition occasioned by an external attack.

6.4 Summary

In this chapter, we covered the implementation and evaluation of our FoCuS. Specifically, we first introduced the development environments and tools such as the Vert.x and Maven. Then, we elaborated the implementation details of the IAM, BIdP, and dashboard modules. Relying on these module implementations, we built a docker image, deployed docker containers, reproduced case studies proposed in chapter 3, and presented them using the dashboard GUI.

More importantly, we analyzed the performance of our FoCuS from the perspective of scalability, interoperability, mobility, storage cost, transaction fee, and delay time in the real Bitcoin network, computation and communication cost, and lightweight verification overheads. We also tested the threat detection module in a smart home testbed whose experimental results show that the intrusion detection module can effectively detect both known and unknown threats with high detection rates and low false positive alarms in cyber-physical devices.

Although illustrating the device management module in the FoCuS microservice architecture (see, Figure 6.2), we only use JSON files to simulate devices in the current implementation in that the device management mainly involves drivers or adapters for connecting real devices compared with security aspects. However, the device management module is still essential to connect the threat detection module and the IAM module as shown in Figure 6.2. Besides, the current threat detection module is independently tested and not integrated into the FoCuS due to the device management module. Therefore, the device management module will be our next priority in order to integrate our threat detection module deeply.

CHAPTER 7

Conclusion

Contents

7.1 Summary of the Contributions

The premise of the Internet of Things (IoT) is to interconnect not only sensors, mobile devices, and computers but also individuals, homes, smart buildings, and cities, as well as electrical and water grids, automobiles, and airplanes, to mention a few. However, realizing the extensive connectivity of IoT while ensuring user security and privacy still remains a challenge. In this thesis, we propose an IoT secure infrastructure called FoCuS which focuses on security fundamental concepts (namely, the authentication and the authorization). Taking into account intrinsic IoT unconventional characteristics such as scalability, heterogeneity, dynamic changes, and limited resources, we take advantage of the blockchain technology, social IoT, SOA paradigm, asynchronous and non-blocking concepts to build the user-centric IoT secure infrastructure. Our proposed infrastructure comprises two pillars: the blockchain-based identity management framework for IoT (BIMSIT) and the decentralized IoT access control framework (DITAC). We briefly summarize our contributions as follows:

- **Hierarchical Identity Information Management.** In the BIMSIT framework, we propose a hierarchical identity structure to organize all identities of a subject (i.e., individuals or collectives). Compared to LDAP directories, the hierarchical representation adds a semantic on how identities are structured (functional vs. organizational). Besides, the concept of digital identity is generalized to cover things (physical and virtual things, such as devices, files, processes). Particularly, the identity is extended to identify services (APIs) as the finest granularity of identifiable things. Finally, the

management of identity is relatively simple, enabling easy user-centric management to handle individual and collective at any depth of representations simultaneously.

- **Distributed Blockchain-based Identity Provider.** Compared to traditional identity management systems, the adopting of the blockchain technology to generate identities eliminates the dependency on third-party identity providers and advocates for user-centric security, independent and neutral networks of identities for IoT. Through the blockchain technology, we build a trustless, decentralized, immutable and public ledger that records users and organizations identities, and facilitates the development of new authentication and authorization mechanisms without relying on third-party identity providers.
- **Universally Social Model for IoT.** The core idea behind the FoCuS infrastructure is building the IoT as social P2P networks to ensure decentralization and sustainable growth, and the universally social model could unify all IoT entities from people, companies, organizations to devices, applications, and services. Three types of relationships ($S2S$, $S2T$, $T2T$) not only help to solve the scalability, heterogeneity problems but also build user-centric security solutions. Relationships among subjects make it possible for individuals or collectives to freely join a social network and interact with each other, which thus could form a scalable network at a large scale. Relationships among IoT things are the key to unify heterogeneous things. Relationships between subjects and IoT things allow owners to set security policies to secure their own devices or services, which could integrate all IoT things to subjects with built-in security policies through ownership.
- **User-centric Access Control.** Taking advantage of these relationships, the DITAC framework makes it possible to build trust in trustless IoT environments and then grants permissions via mapping users from a subject or among subjects. The relationship-based access controls not only allow owners to set their access control policies without relying on the authentication of third parties but also incorporate many technical details in other access control models such as attribute-based access controls and role-based access controls, which makes users directly grant permissions using a human-readable policy language. For example, Alice allows Claire (role: a housekeeping) to open Alice's door lock from 1 p.m. to 2 p.m (attribute: a time-attribute restriction).
- **Autonomic Threat Detection at Runtime.** In order to prevent malicious attacks at runtime, we also build a threat detection system for our decentralized access control framework, which can create a reference model for each smart object that describes its normal behavior. The autonomic threat detection capabilities in the DITAC framework enables IoT security as a continuum between runtime execution and access control sessions by preventing malicious attacks and abusive behaviors from breaching smart devices and stealing sensitive information.

Generally speaking, the design of FoCuS identifies all entities in IoT realm without relying on unnecessary third-party authorities via the blockchain-based identity management framework. It not only eliminates dependence on third

parties who compromise privacy but also avoids phishing and single point of failure from the traditional identity solutions. Furthermore, in order to formally verify our BIMSIT framework, we define the threat model, evaluate key compromise, verify the correctness of the simplified authentication protocol using the BAN logic, and analyze security and privacy properties from the theoretical perspective. More importantly, our proposed blockchain-based identity management framework is scalable (namely, the mean delay time T does not increase as the number of identity provider nodes increases) and feasible in resource-constrained devices according to our analysis in chapter 6.

Relationships between IoT entities form a universally social model that unifies all IoT entities from people, companies, organizations to devices, applications, and services. The adoption of social relationship concept in IoT and the global blockchain-based IdP make the IoT extensible and ubiquitous networks in which all heterogeneous social entities (namely, individuals, collectives, things) can freely join and leave based on created hierarchical identities. More importantly, relationships from socialized IoT can be used to design user-centric access control framework in trustless peer-to-peer networks. In other words, owners can set security policies to secure their own devices or services through ownership without relying on third-party authorities. Furthermore, in order to enhance the availability of our FoCuS, an autonomic threat detection mechanism is introduced as the complementary of our access control framework. It can continuously monitor anomaly behaviors of device or service operations and feed the result to our access control framework.

In our prototyping, FoCuS is shaped into reusable software components that provide reusable security services, including identification services, relationship management services, authentication services, access control services, and threat detection services. These services are defined and developed through a set of universal REST APIs, which are loosely coupled, reusable and composable components for any scenario in the heterogeneous IoT environments. Moreover, the adoption of the event-driven model in our implementation makes our FoCuS deployed on edge nodes capable of handling high concurrency, real-time, and dynamic request in IoT environments. The theoretical analysis and performance evaluation results demonstrate that our proposed IoT secure infrastructure – FoCuS, can deal with previously mentioned identity and access control challenges (namely, *IDC-A, IDC-B, IDC-C, ACC-A,* and *ACC-B*), and provide security or privacy services in IoT environments.

7.2 Future Research Directions

However, there are still many aspects that need to be further explored to improve the entire infrastructure for IoT. In this section, we identify and analyze some future research directions.

7.2.1 Privacy

Privacy preserving refers to the protection of users' sensitive information such as identity information, location, mobility traces, habits from any other parties. From the perspective of users, privacy preserving includes two aspects:

- identity information protection from identity providers
- sensitive application data protection from service providers

Many works [EE12; Hua+12; WW11] in academic papers and IT industry are proposed to preserve the sensitive application data rather than identity information stored in identity providers. In most cases, identity providers and service providers are bound together, and they require some personal information in order to authenticate users. For instance, users may protect their location information from map service providers by disabling the location service. However, they ignore the leakage of their personal identity information by identity providers exposed to security vulnerabilities. Albeit these proposed solutions in [EE12; Hua+12; WW11] solve the privacy problem to some extent, their identity information is still exposed to identity providers.

Before blockchains, users' privacy is difficult to be fully preserved owing to the existence of centralized identity providers in that the identity information protection remains unsolved. Service accessors and service providers need to grant full trust to their identity providers. In other words, centralized identity providers could track activities between service accessors and service providers, which compromises the identity information privacy. Fortunately, the self-sovereign blockchain identity management is recently taking the control right of identities back to end-users from the third-party identity providers. The design of identity solutions is thus subject to a paradigm shift by which users decide to whom their sensitive personal information could be revealed (from user's perspective) instead of trusting identity providers to manage their personal information. Although users could have full control over their personal information in blockchain-based identity management systems, the public blockchains can still expose some identity information. For instance, the endorsement from blockchain root identities is required by partial identities from time to time, which increases the risk of being compromised. Therefore, we still need to introduce privacy preserving schemes in that the introduction of privacy-preserving tools such as zero-knowledge proofs [GMR89] could bring the selective disclosure of sensitive personal information and perfect online identity privacy into reality.

The interactive zero-knowledge proofs are extensively studied in designing secure authentication schemes for IoT. Compared to the complex key management of symmetric-key based authentication solutions, and large memory and communication overhead of public-key based authentication solutions, interactive zero-knowledge proofs are relatively efficient cryptography schemes in authentication and key exchange. For instance, Ma et al. [MGZ14] proposed a zero-knowledge proof based authentication protocol (TinyZKP) for sensor nodes in wireless body area networks. Potop-Butucaru et al. [KPC16; BP18] presented a more efficient authentication solution compared with TinyZKP, which is based on zero-knowledge proof and cryptographic commitment schemes. The zero-knowledge proof allows two parties to verify their identities through several

interactive messages without revealing new knowledge about shared secrets while the cryptographic commitment scheme is used to prevent replay attack in the proposed solution. Privacy preserving has always been one of the hottest topics in the blockchain industry. Since 2009, academic researchers have proposed many privacy-preserving schemes including Pinocchio [Par+13], zk-Garbled Circuits [JKO13], zk-SNARKs [Ben+13], ZKBoo [GMO16], zk-STARKs [Ben+17], and Bulletproofs [Bün+18]. The goal of these schemes is to design a succinct non-interactive zero-knowledge protocol to protect the privacy in blockchain systems. However, due to the complexity of these schemes, only zk-SNARKs and Bulletproofs are deployed in Zcash [Sas+14] and Monero [Noe15] systems, respectively. As for the integration with blockchain smart contracts, it still needs many efforts from academia and industry.

To sum up, whether it is an interactive or a non-interactive zero-knowledge protocol solution, their purpose is identical, that is, transactions or services can be delivered while fully ensuring the privacy of users. Therefore, in order to reduce the risk bought by the endorsement from blockchain root identities in our solutions, corresponding privacy-preserving techniques need to be developed.

7.2.2 Trust

The trust management is closely related to the identity and access control management framework. Although there is no consistent definition on the concept of trust [Sic+15], researchers recognize the importance of trust management. Many schemes have been proposed to manage trust in the context of IoT in order to deal with misbehaving IoT devices. Chen et al. [Che+11] presented a fuzzy reputation-based trust management solution for IoT wireless sensor networks, in which they considered the packet forwarding/delivery ratio and energy consumption as the Quality of Services (QoS) metrics to evaluate the trust relation. Recently, the SIoT [Atz+12] obtains much attention due to the high extensibility through the integration of social networks concept into IoT. Therefore, many trust management solutions have emerged based on the SIoT paradigm. For instance, authors in [Nit+12; NGA14; CBG16] introduced trust management solutions for the SIoT paradigm. Authors in [BC12] presented a similar trust management protocol for IoT with no centralized trusted authority. All these solutions make use of not only the QoS metrics but also the ownership between devices and owners as well as the social relationship between users to evaluate the trust to defend against attacks.

However, these trust management solutions are built on the previous implicit identity assumption that users and service providers should put all trust to their identity providers so that they could identify each other in the same security domain. Within the same security domain, users and service providers trust and rely on the same identity provider, admitting that their personal information will not be compromised or exploited by the identity provider or third parties. In many cases, identity providers are subject to vulnerabilities which expose personal information repositories to be stolen by deliberated attackers (e.g., Equifax data breach [CSP17] and Facebook security breach [IF18]). However, the implicit trust in identity providers becomes questionable with the increasing attacks [CSP17;

IF18]. Undoubtedly, blockchain-based identity management systems eliminate unnecessary information exposure to third parties and provide many excellent characteristics such as immutability, neutrality, and secure timestamping, which could be used for building trust relationships. For instance, Lu et al. [Liu+17] presented an interesting approach to building the trust reputation via tailored Ethereum tokens. Zhu [ZB18] combined the blockchain and the social networks between all IoT entities to build a security architecture for IoT, which lays a solid foundation for trust management. However, these decentralized or distributed ways still face up to many difficulties in building the reputation system or feedback mechanism for aggregating trust relations on each party including all subjects and service providers. For instance, how to design decentralized reputation systems and prevent from colluding and malicious adversaries. Fortunately, some multi-party computation techniques shed light on building decentralized privacy preserving reputation systems [Mou+15]. For example, Hasan et al. [HBB12; Has+13] proposed their decentralized preserving privacy reputation systems under semi-honest and dishonest adversarial models respectively, in which they combine addictive homomorphic encryption and design a K-shares reputation protocol with capabilities of preventing malicious adversaries from disrupting the proposed reputation protocol.

The FoCuS infrastructure seeks to be a Fog-based secure infrastructure designed for IoT systems. However, realizing Fog computing still has a long way to go, especially when it comes to modules such as hardware virtualization, load balancing, location services. Therefore, extending the FoCuS infrastructure and its security modules with other Fog computing modules is an interesting challenge in order to put our infrastructure into a final product. Besides, the mnemonic code words are used to easily backup keys in the BIMSIT framework. However, the protection of these words is critical. Although these words are offline and could be written down on papers, new protection mechanisms like multi-party computation, secret sharing or social recovery, are still need to be explored to provide more reliable and resilient backup solutions. Also, the established relationships in the DITAC framework are too primitive to express complex access control scenarios such as multi-subject with transitive relationships. In addition, the current threat detection mechanism is designed for the physical perception layer and hence detecting attacks by analyzing the behavior of other IoT layers is conducive to the completion of runtime IoT threat detection module. For example, if an attacker collects enough sensors' information (e.g., one day of information), it can launch a replay attack without the need of using the same data set. Therefore, we are working on several solutions for such kind of attacks, including big data analytics to search for patterns in the stored data, and moving target defense to change configurations, making it extremely difficult for an attacker to collect relevant data. In order to deeply integrate the threat detection into the access control framework, some reactive and adaptive security monitoring mechanisms [ERM18] are indispensable to help decision-making based on monitoring results in threat detection systems.

APPENDIX A

The Scalability Mathematics Analysis

Contents

A.1 Introduction

Our IoT security infrastructure mainly consists of two frameworks: the Blockchain based Identity Management Framework for IoT (BIMSIT) and the Decentralized IoT Access Control (DITAC) Framework. Since the DITAC is decentralized and could be configured by each subject without the collaboration with others, the scalability of our IoT security solution refers to the scalability evaluation of the distributed BIMSIT.

A.2 Scalability Metric Definition

Assumption 3 *Most of the transactions are identity lookup operations, which are free offline transactions without fees in permissionless blockchain platforms like Ethereum or Bitcoin.*

In order to evaluate the scalability of the BIMSIT, we still need to make this assumption which agrees with our intuition. On chain identity operations (i.e., registration, revocation, and update) are relatively rare compared to the off-chain identity lookup operations. Usually, users do not need to modify their identities after registering their identity without losing their key information. For instance, we rarely modify the password of our social media like Facebook or Google unless

they forget their passwords or attackers acquire their accounts.

According to the scalability evaluation theory for heterogeneous and distributed systems proposed in [JW00], we could quantify the scalability of the BIMSIT using the proposed productivity based scalability metric.

Definition 11 *In BIMSIT system, we define the productivity function of evaluating the performance and the cost of the system operation as follows:*

$$F(N) = \lambda(N) \times \frac{Q(N)}{C(N)} \tag{A.1}$$

- N is the number of nodes with the full identity repository in the networks.
- $\lambda(N)$ stands for the rate of providing valuable services, which means the throughput of the BIMSIT system.
- $Q(N)$ indicates the quality of services QoS, which is based on the ratio between the mean delay T and the target delay $\hat{T}$ through the function $f(T/\hat{T})$.
- $C(N)$ refers to the total cost of providing the services, which means resource usage of providing a certain number of identity lookup operations under the Assumption 3 including CPU, RAM, I/O workload and Storage Consumption.

Definition 12 *BIMSIT scalability. The scalability metric for BIMSIT when the network scale (specifically, the number of nodes with the full identity repository in the networks) varies from State 1 to State 2 is defined:*

$$\psi(S_1, S_2) = \frac{F(S_2)}{F(S_1)} \tag{A.2}$$

In BIMSIT system, with the increment of the full identity nodes (N), productivity function F is proportional to λ/C. For instance, if increasing the same number of nodes, the throughput and the cost of the BIMSIT system will turn into $2 * \lambda$ and $2 * C$ respectively. If combining with the f function proposed in [JW00],

$$f(\frac{T}{\hat{T}}) = 1/(1 + (\frac{T}{\hat{T}})^n), n \geq 1 \tag{A.3}$$

we could manipulate the equation A.2 as follows:

$$\psi(S_1, S_2) = \frac{\lambda_2 \cdot C_1 \cdot (T_1^n + \hat{T}^n)}{\lambda_1 \cdot C_2 \cdot (T_2^n + \hat{T}^n)}, n \geq 1 \tag{A.4}$$

A.3 Scalability Analysis Proof

Theorem 1 *The BIMSIT is scalable from S_1 to S_2 as the increment of the number of the full identity nodes in the networks.*

Proof: Suppose the scalable factor of the number of the full identity nodes (N) is k, then we have $\lambda_2 = k\lambda_1$, $C_2 = kC_1$, $T_2 = T_1/k$. After the manipulation of the equation A.4, we could have:

$$\psi(S_1, S_2) = 1 + \frac{(k-1)T_1^n}{T_1^n + k\hat{T}^n}, \lim_{k\to\infty} \psi(S_1, S_2) = 1 + \frac{T_1^n}{\hat{T}^n}, n \geq 1 \tag{A.5}$$

Therefore, when the mean delay of the BIMSIT system reaches the designed target delay time (namely, $T_1 \approx \hat{T}$), the value of ψ has a bounded increase from 1 to around 2 with the increment of network scale to infinity, which demonstrates the BIMSIT system has positive scalability. In other words, the more nodes with full identity repository, the smaller the mean delay becomes.

APPENDIX B

IAM Module Full Grammar Implementation

```
IAM:= SUBJECTS
SUBJECTS := SUBJECT |  SUBJECTS SUBJECT
SUBJECT:= subject <subjectname> “,”  hierarchy HIS

HIS:= seed <seedstring> “,” MASTERKEY “,” ROOT_IDENTITY
MASTERKEY:=  KEYPAIR

ROOT_IDENTITY := IDENTITY_BODY “,” partial_identities(PARTIAL_IDENTITIES) “,”
                    map(ONLINE_ROOTKEY “,” OFFLINE_ROOTKEY) “,”  DOMAINS

PARTIAL_IDENTITIES:= PARTIAL_IDENTITY
                     | PARTIAL_IDENTITIES  “,” PARTIAL_IDENTITY
PARTIAL_IDENTITY:=  IDENTITY_BODY “,” map(ONLINE_KEY “,” OFFLINE_KEY) “,”
                    DOMAINS | IDENTITY_BODY “,”
                    partial_identity(PARTIAL_IDENTITIES) “,”
                    map(ONLINE_KEY “,” OFFLINE_KEY) “,” DOMAINS

ONLINE_ROOTKEY:= KEYPAIR
OFFLINE_ROOTKEY:= KEYPAIR
ONLINE_KEY:= KEYPAIR
OFFLINE_KEY:=  KEYPAIR

KEYPAIR := < level:<path> “,”  sk STRING  “,”  pk STRING “,”

IDENTITY_BODY:= identity <identityName> “,” identifier <did> “,”
                credentials CREDENTIALS “,” attributes JSON_OBJECT

CREDENTIALS:= KEYPAIR

DOMAINS := DOMAIN |  DOMAINS “,” DOMAIN
DOMAIN:= domain <domainName> “,” dominance ROLES

ROLES := ROLE  |  ROLES “,” ROLE
ROLES: = role <role_name> services “:” SERVICES “,”
```

```
SERVICES := SERVICE |  SERVICES “,” SERVICE

RELATIONSHIP:= HEADER “,”  PAYLOAD

HEADER:= id <relationshipIdentifer> “,” label < relationshipLabel> “,”
         timestamp “<timestamp> ‘,” version <version> “,”
              subjects <subject1RootIdentifier> <subject2RootIdentifier>
          SIGNATURES “,” created <creationDate> “,” expired <expiredDate>
          “,” integrity <HashOf Payload >
SIGNATURES := signatures<HeaderSignedBySubject1> <HeaderSignedBySubject2>
                                | signature <HeaderSignedBySubject1>

PAYLOAD:= claims “:” JSON_OBJECT “,” CAPABILITIES

CAPABILITIES:= CAPABILITY |  CAPABILITIES “,” CAPABILITY
CAPABILITY:= “(“ <subject1PartialIdentifier> “,”
                     < subject2PartialIdentitfier> “)”

OPERATIONS: [register | dispose] OBJECT
OBJECT:= name <object_urn> “,” identifier <partialIdentity.Identifier> ”,”
     access <uniformResourcelocator> “,” owner <RootIdentity.identifier>
     “,” SERVICES

SERVICES:= SERVICE | SERVICES “,” SERVICE
SERVICE:= name <service_name> url < uniformResourcelocator > input INPUT*
       output OUTPUT*

POLICY:= RULES “,” updated <date>
RULES := RULE | RULE “,” RULES

RULE:= on [<domain_name> | <role_name> | <relationshipIdentifer> |
<object_name> | <service_name>] [accept | deny] [ where SECURITY_CONTEXT ]

SECURITY_CONTEXT:= [ <role_attribute> LOGIC_OPERATOR <role_value> |
          <relationship_attribute> LOGIC_OPERATOR <relationship_value> |
           <context_attribute> LOGIC_OPERATOR <context_value> ] +

STRING := [aA-zZ]
JSON_OBJECT:= json object such as {<key = value>}
SEPERATOR_SEMICOLON := “,”
SEPERATOR_COLON := “:”

NB: All leaf nodes are (partial-)identities associated to services
```

APPENDIX C

Screenshots of Implementations

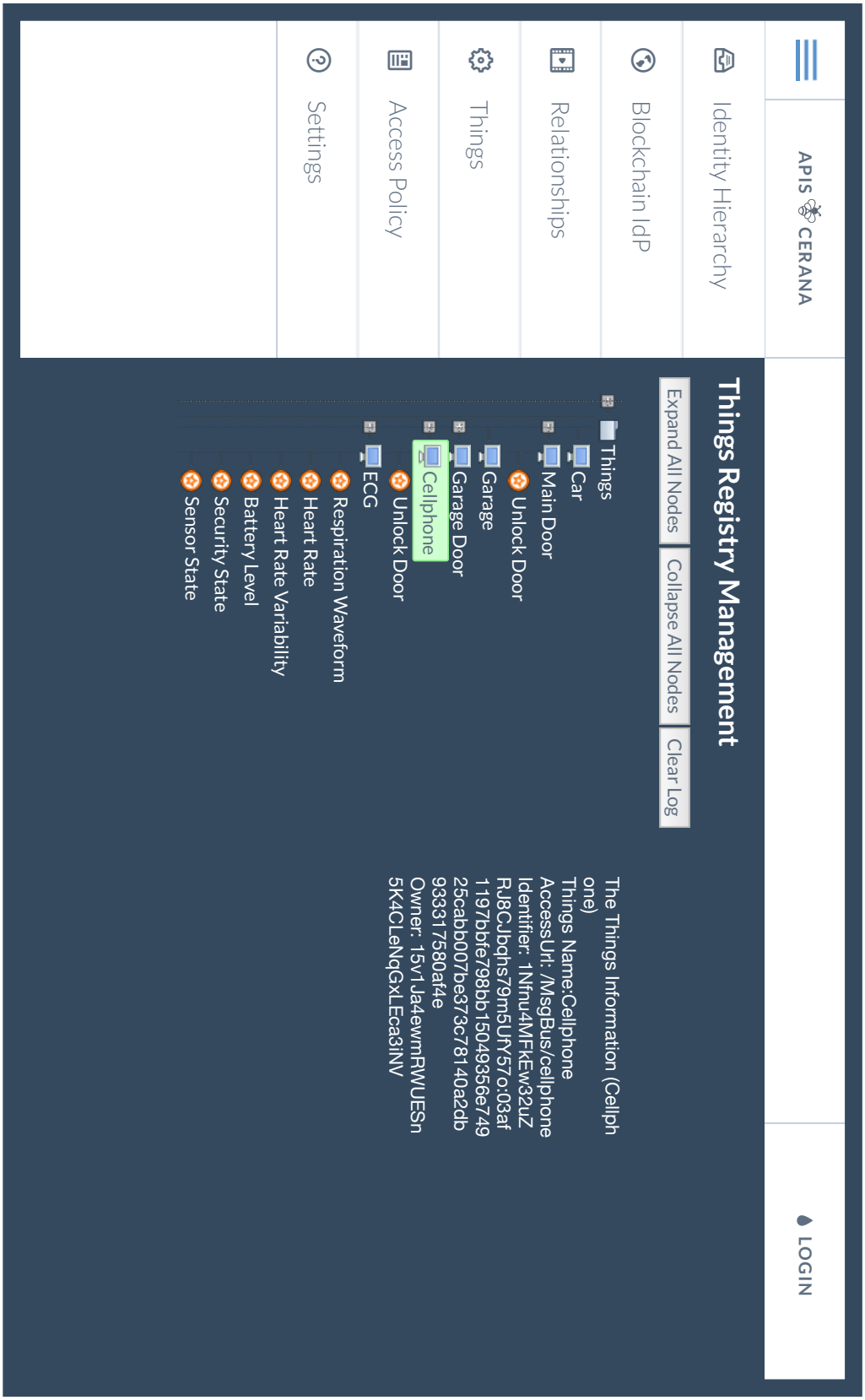


Figure C.1 – Alice's things tree

Figure C.2 – Alice's BIdP driver

Figure C.3 – Bitcoin BIMSIT dashboard

Figure C.4 – Access policy page

Figure C.5 – Received establishing relationship request

Figure C.6 – Claire's confirmed relationship

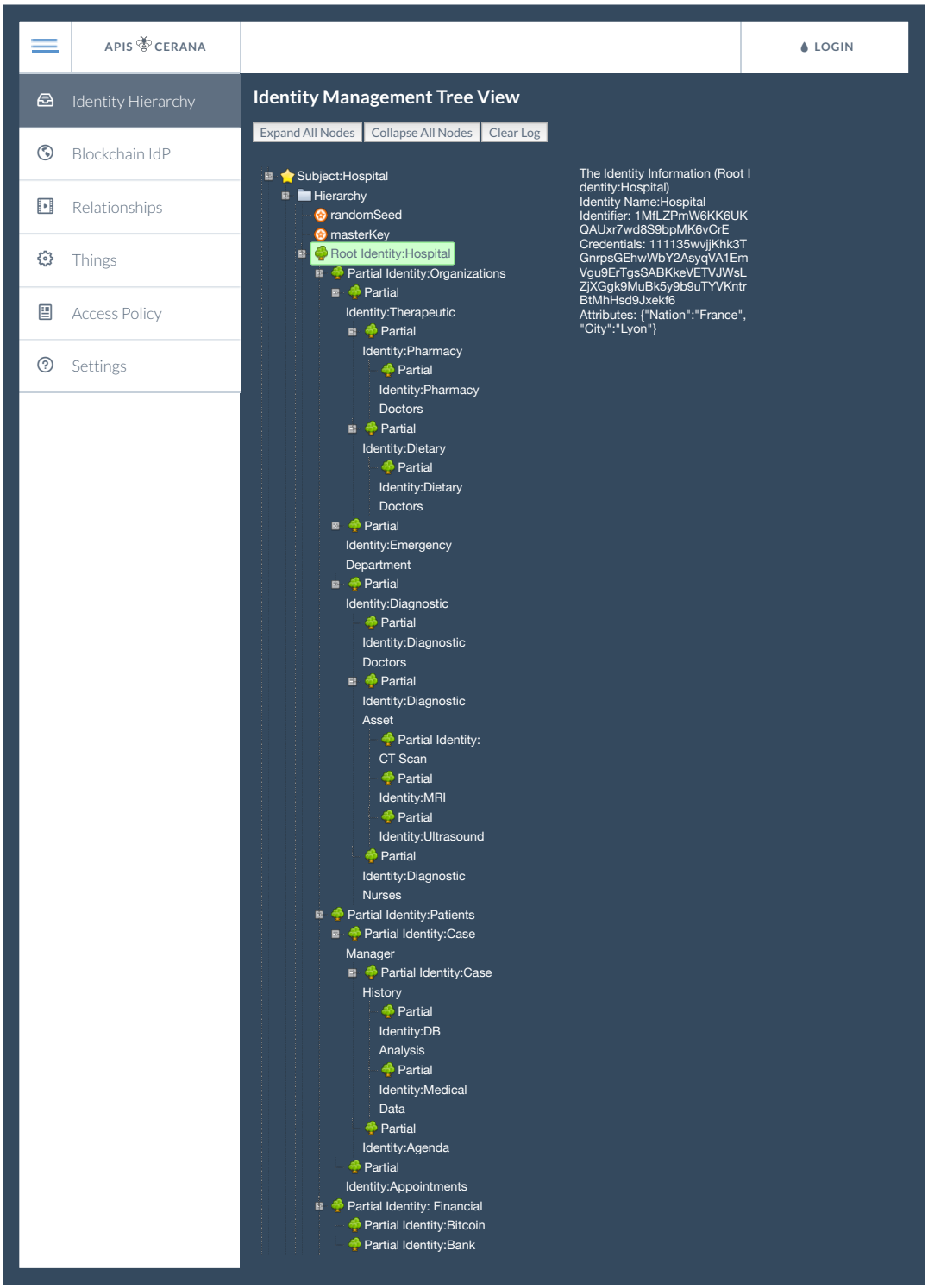


Figure C.7 – Hospital identity tree

APPENDIX D

Diagram of the FoCuS Infrastructure

Figure D.1 describes the diagram of the identity, relationship management, authentication, and authorization respectively. In the diagram, there are two subjects (i.e., Alice and Claire) and a smart lock that belongs to Alice. We illustrate the diagram as follows:

Step 1: Alice provides names and attributes to generate her identity set (IDS definition: see Equation 4.15) through

$$IDS = generateIDS(name, attributes) \tag{D.1}$$

Step 2: After having the IDS, Alice constructs identity blockchain transactions through the $generateIdentityTransaction(txType, txConstructPara)$, where the $txType$ points out the type of the constructing identity transaction (i.e., registration, update, or revocation) and the $txConstructPara$ is the required parameters (e.g., identifier, public keys, and signatures of the transaction using private keys) to construct the specific transaction.

Step 3: Then, Alice broadcasts the constructed transaction (TX) through the $broadcastIdentityTransaction(TX)$ to the P2P network. Miners in the P2P networks will verify the transaction through the $verifyIdentityTransaction(TX)$ and seal the transaction into a block. Once having enough mined blocks after the block, others can confirm that the identity has reached consensus.

Step 4: Using the $generateIdentityAssertion(sk_P, pk_V)$, Alice, the Prover ($P$), takes the private key of P and the public key of the smart lock, the Verifier (V), to produce an Identity Assertion (IA), which refers to a verifiable claim, denoted by

$$IA = \{\{N_P, K_{P,V}\}_{sk_P}\}_{pk_V} \tag{D.2}$$

where the nonce and symmetric key generated by P, are signed using P's private key and then encrypted with the public key of V.

Step 5: P requests a service through the $requestSrv(IA, P.identifier, Service)$ which takes the IA generated by P, the identifier of P, and the corresponding service provided by V.

Step 5.1: V retrieves the public key in the blockchain ledger using P's identifier through the $identityLookup(P.identifier)$.

Step 5.2: In order to verify P's identity, V decrypts the IA with V's private key and then check the signature using P's public key through the $verifyIdentity(IA)$.

Step 5.3: V generates IA to prove the identity of V using the previous $generateIdentityAssertion(sk_V, pk_P)$.

Step 5.4: V gives the service response requested by P, coupled with the IA of V through the $responseSrv(serviceResponse, IA)$, where $serviceResponse$ is the result of P's request service.

Step 6: For relationships, Claire can establish a relationship with Alice through the $establishRelationshipRequest(IA, identifier, relEstablishPara)$, where the IA refers the identity proof of Claire, the $identifier$ is Claire's identifier, and the $relEstablishPara$ is the required parameters (e.g., label, identifiers, accessPoint, Service, and Capability) to establish the relationship.

Step 6.1: Alice retrieves the public key in the blockchain ledger using Claire's identifier through the $identityLookup(Claire.identifier)$.

Step 6.2: Alice verifies Claire's identity through the $verifyIdentity(IA)$.

Step 6.3: Alice checks the relationship establishment request from Claire and gives the expired time of the relationship if Alice agrees on establishing the relationship through the $establishRelationshipResponse(expiredTime)$.

Step 7: For authorization, Alice can create roles that carry privileges through the $updateServicePolicies(roleName, services, constraints)$, where the $services$ point out the accessible services and the $constraints$ refer to limitations when granting the privileges.

Step 8: After that, Alice can set corresponding policies to the smart lock through the $configServicePolicies(target, policies)$, where the $target$ designates the device and $policies$ are access control permissions set by the owner.

Step 9: When Claire requests a service (e.g., open the lock) through the previous $requestSrv(Claire.IA, Claire.identifier, Service)$, the lock or Alice (owner) firstly verifies the identity of Claire through $verifyIdentity(IA)$. If the identity is valid, the lock continues to check the policies prescribed by Alice (owner) through $checkPolicies(requestPermissions)$. According to prescribed permissions, the lock gives the response through the previous $responseSrv(serviceResponse, IA)$.

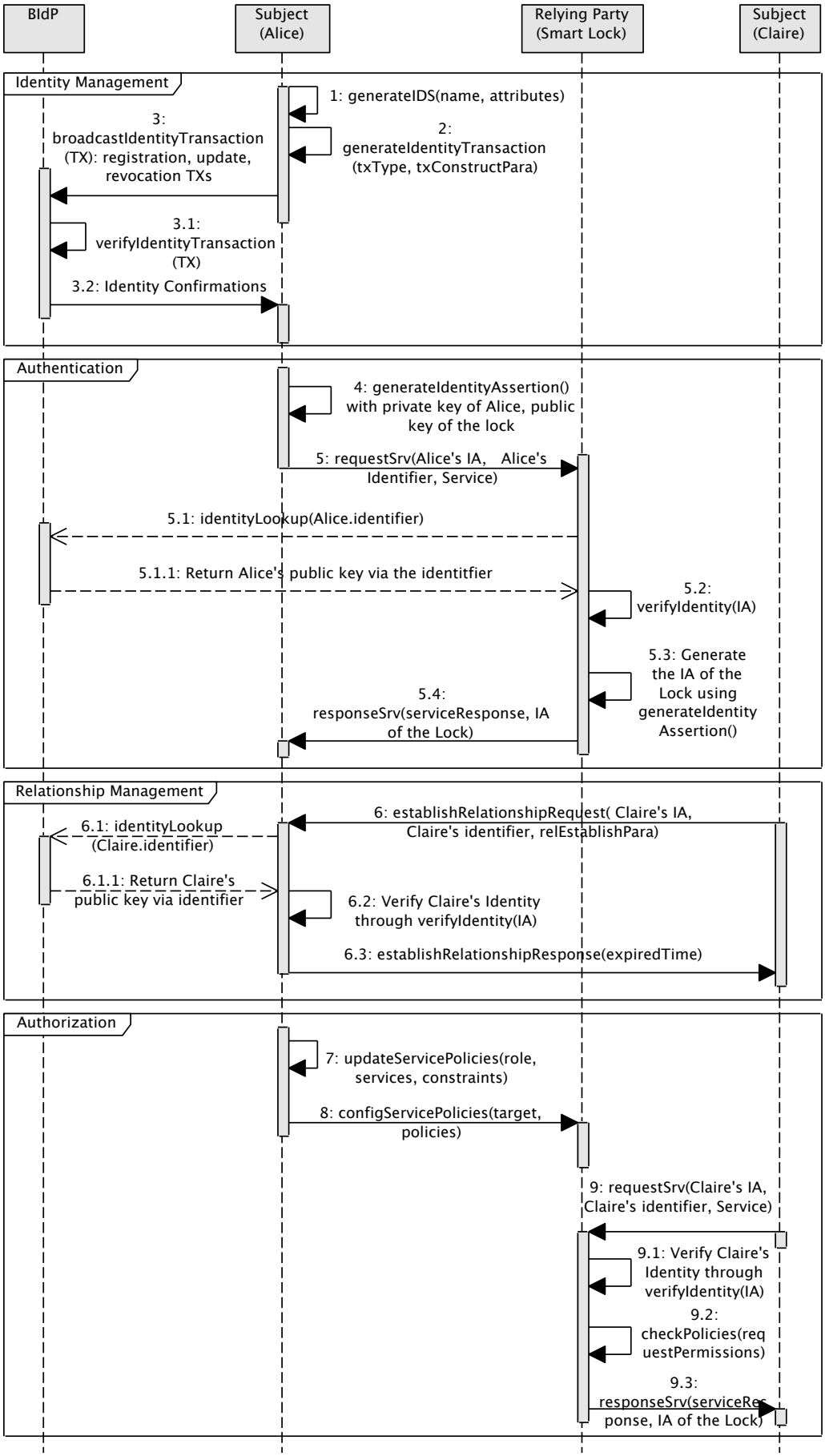


Figure D.1 – Diagram of the identity, relationship management, authentication, and authorization

Bibliography

[ABM17] Sarah Azouvi, Mustafa Al-Bassam, and Sarah Meiklejohn. "Who Am i? Secure Identity Registration on Distributed Ledgers." *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2017, pp. 373–389.

[AHS06] Karl Aberer, Manfred Hauswirth, and Ali Salehi. "A Middleware for Fast and Flexible Sensor Network Deployment." *Proceedings of the 32nd International Conference on Very Large Data Bases*. VLDB Endowment, 2006, pp. 1199–1202.

[AHS11] Gergely Alpár, Jaap-Henk Hoepman, and Johanneke Siljee. "The Identity Crisis. Security, Privacy and Usability Issues in Identity Management." *arXiv preprint arXiv:1101.0427*, 2011. URL: http://arxiv.org/abs/1101.0427 (visited on 10/19/2016).

[AIM14] Luigi Atzori, Antonio Iera, and Giacomo Morabito. "From Smart Objects to Social Objects: The next Evolutionary Step of the Internet of Things." *IEEE Communications Magazine*, vol. 52, no. 1, 2014, pp. 97–105.

[AKS09] Gail-Joon Ahn, Moonam Ko, and Mohamed Shehab. "Privacy-Enhanced User-Centric Identity Management." *Proceedings of the 2009 IEEE International Conference on Communications*. IEEE, 2009, pp. 1–5.

[Akt+13] Evangelos Aktoudianakis et al. "Policy Templates for Relationship-Based Access Control." *Proceedings of the 2013 Eleventh Annual International Conference on Privacy, Security and Trust*. IEEE, 2013, pp. 221–228.

[Ala+15] Mahdi Ben Alaya et al. "Toward Semantic Interoperability in oneM2M Architecture." *IEEE Communications Magazine*, vol. 53, no. 12, 2015, pp. 35–41.

[Ali+16] Muneeb Ali et al. "Blockstack: Design and Implementation of a Global Naming System with Blockchains," 2016. URL: https://pdfs.semanticscholar.org/2434/f6773677316d1ac9f2e1d8e2fcb648ee484c.pdf?_ga=2.63072687.1887205232.1557575790-1047492284.1553888816 (visited on 06/03/2016).

[All16] Allseen Alliance. "Alljoyn Framework." *Linux Foundation Collaborative Projects. URl: https://allseenalliance. org/framework (visited on 09/14*, 2016.

[Alt] Altair Engineering. *Carriots - Internet of Things Platform | Home*. URL: https://www.carriots.com/ (visited on 08/24/2018).

[Ama] Amazon. *Amazon Web Services IoT Applications & Solutions*. URL: https://aws.amazon.com/iot/ (visited on 08/22/2018).

[Ang+10] Pelin Angin et al. "An Entity-Centric Approach for Privacy and Identity Management in Cloud Computing." *Proceedings of the 29th IEEE Symposium on Reliable Distributed Systems*. IEEE, 2010, pp. 177–183.

[Ant14] Andreas M. Antonopoulos. *Mastering Bitcoin: Unlocking Digital Cryptocurrencies*. O'Reilly Media, Inc., 2014.

[APS16] Tahmina Ahmed, Farhan Patwa, and Ravi Sandhu. "Object-to-Object Relationship-Based Access Control: Model and Multi-Cloud Demonstration." *Proceedings of the 2016 IEEE 17th International Conference on Information Reuse and Integration*. IEEE, 2016, pp. 297–304.

[Arj+12] Pandarasamy Arjunan et al. "Sensoract: A Privacy and Security Aware Federated Middleware for Building Management." *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. ACM, 2012, pp. 80–87.

[Arj+15] Pandarasamy Arjunan et al. "SensorAct: A Decentralized and Scriptable Middleware for Smart Energy Buildings." *Proceedings of the 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*. IEEE, 2015, pp. 11–19.

[Arm+08] Alessandro Armando et al. "Formal Analysis of SAML 2.0 Web Browser Single Sign-on: Breaking the SAML-Based Single Sign-on for Google Apps." *Proceedings of the 6th ACM Workshop on Formal Methods in Security Engineering*. FMSE '08. New York, NY, USA: ACM, 2008, pp. 1–10.

[ASP17] Tahmina Ahmed, Ravi Sandhu, and Jaehong Park. "Classifying and Comparing Attribute-Based and Relationship-Based Access Control." *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*. ACM, 2017, pp. 59–70.

[Atz+12] Luigi Atzori et al. "The Social Internet of Things (Siot)–When Social Networks Meet the Internet of Things: Concept, Architecture and Network Characterization." *Computer Networks*, vol. 56, no. 16, 2012, pp. 3594–3608.

[Aug+17a] Daniel Augot et al. "A User-Centric System for Verified Identities on the Bitcoin Blockchain." *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Lecture Notes in Computer Science. Springer, Cham, 2017, pp. 390–407.

[Aug+17b] Daniel Augot et al. "Transforming Face-to-Face Identity Proofing into Anonymous Digital Identity Using the Bitcoin Blockchain." *arXiv preprint arXiv:1710.02951*, 2017.

[AWSa] AWS. *Amazon FreeRTOS - IoT Operating System for Microcontrollers*. URL: `https://aws.amazon.com/freertos/` (visited on 08/23/2018).

[AWSb] AWS. *AWS Greengrass - Amazon Web Services*. URL: `https://aws.amazon.com/greengrass/` (visited on 08/23/2018).

[AWSc] AWS. *AWS IoT 1-Click Overview - One Click Creation of an AWS Lambda Trigger for Any Device*. URL: `https://aws.amazon.com/iot-1-click/` (visited on 08/23/2018).

[AWSd] AWS. *AWS IoT Button Overview - Cloud Programmable Dash Button*. URL: `https://aws.amazon.com/iotbutton/` (visited on 08/23/2018).

[AWSe] AWS. *AWS IoT Core Overview - Amazon Web Services*. URL: `https://aws.amazon.com/iot-core/` (visited on 08/23/2018).

[Axo15] Louise Axon. "Privacy-Awareness in Blockchain-Based PKI," 2015.

[Ban+11] Soma Bandyopadhyay et al. "A Survey of Middleware for Internet of Things." *Recent Trends in Wireless and Mobile Networks*. Ed. by Abdulkadir Özcan, Jan Zizka, and Dhinaharan Nagamalai. Vol. 162. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 288–296.

[BAN89] Michael Burrows, Martin Abadi, and Roger M. Needham. "A Logic of Authentication." *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. Vol. 426. The Royal Society, 1989, pp. 233–271.

[Bas17] Mustafa Al-Bassam. "SCPKI: A Smart Contract-Based PKI and Identity System." *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*. ACM, 2017, pp. 35–40.

[BC12] Fenye Bao and Ing-Ray Chen. "Dynamic Trust Management for Internet of Things Applications." *Proceedings of the 2012 International Workshop on Self-Aware Internet of Things*. ACM, 2012, pp. 1–6.

[Bel14] Charles Bell. *Beginning Sensor Networks with Arduino and Raspberry Pi*. Apress, 2014.

[Ben+13] Eli Ben-Sasson et al. "SNARKs for C: Verifying Program Executions Succinctly and in Zero Knowledge." *Advances in Cryptology–CRYPTO 2013*. Springer, 2013, pp. 90–108.

[Ben+17] Eli Ben-Sasson et al. "Scalable, Transparent, and Post-Quantum Secure Computational Integrity." *Manuscript.(2017). Slides at https://people. eecs. berkeley. edu/ alexch/docs/pcpip_bensasson. pdf*, 2017.

[BH10] M. Brown and R. Housley. *Transport Layer Security (TLS) Authorization Extensions*. RFC Editor, 2010. URL: `https://www.rfc-editor.org/info/rfc5878` (visited on 08/28/2018).

[Bil13] Stephen A. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. John Wiley & Sons, 2013.

[BIT] BITNATION. *BITNATION: Governance 2.0*. URL: `https://bitnation.co/` (visited on 08/31/2018).

[BKS15] Tuhin Borgohain, Uday Kumar, and Sugata Sanyal. "Survey of Security and Privacy Issues of Internet of Things," 2015. arXiv: `1501.02211 [cs]`.

[Bon+12] Flavio Bonomi et al. "Fog Computing and Its Role in the Internet of Things." *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*. ACM, 2012, pp. 13–16.

[Bor+19] Yogita Borse et al. "Anonymity: A Secure Identity Management Using Smart Contracts." *Available at SSRN 3352370*, 2019.

[BP18] Gewu Bu and Maria Potop-Butucaru. "BAN-GZKP: Optimal Zero Knowledge Proof Based Scheme for Wireless Body Area Networks." *Ad Hoc Networks*, vol. 77, 2018, pp. 28–41.

[BPS16] Smriti Bhatt, Farhan Patwa, and Ravi Sandhu. "An Attribute-Based Access Control Extension for OpenStack and Its Enforcement Utilizing the Policy Machine." *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference On*. IEEE, 2016, pp. 37–45.

[Bra00] Stefan Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. Mit Press, 2000.

[Bru+12] Glenn Bruns et al. "Relationship-Based Access Control: Its Expression and Enforcement through Hybrid Logic." *Proceedings of the Second ACM Conference on Data and Application Security and Privacy*. ACM, 2012, pp. 117–124.

[BSB07] Vittorio Bertocci, Garrett Serack, and Caleb Baker. *Understanding Windows CardSpace: An Introduction to the Concepts and Challenges of Digital Identities*. Addison-Wesley, 2007.

[BSL07] Victor H. Benitez, Edgar N. Sanchez, and Alexander G. Loukianov. "Decentralized Adaptive Recurrent Neural Control Structure." *Engineering Applications of Artificial Intelligence*, vol. 20, no. 8, 2007, pp. 1125–1132.

[BT11] Elisa Bertino and Kenji Takahashi. *Identity Management: Concepts, Technologies, and Systems*. Artech House, 2011.

[Bün+18] Benedikt Bünz et al. "Bulletproofs: Short Proofs for Confidential Transactions and More." *Bulletproofs: Short Proofs for Confidential Transactions and More*. IEEE, 2018.

[BUT14] BUTLER. *uBiquitous, secUre inTernet-of-Things with Location and contExtawaReness*. 2014. URL: `https://cordis.europa.eu/project/rcn/101349_en.html` (visited on 08/19/2018).

[But14] Vitalik Buterin. "A Next Generation Smart Contract & Decentralized Application Platform," 2014. URL: `https://res.tuoluocaijing.cn/20180513220217-hldi.pdf` (visited on 08/19/2018).

[BUT15] BUTLER Smart Gateway. *Open Platforms – sensiNact*. 2015. URL: `http://open-platforms.eu/library/sensinact-aka-butler-smart-gateway/` (visited on 08/19/2018).

[But74] Lampson Butler. "Protection." ACM SIGOPS Operating Systems Review. Vol. 8. 1974, pp. 18–24.

[Car+09] Rafael J. Caro et al. "Smepp: A Secure Middleware for Embedded P2p." *ICT Mobile and Wireless Communications Summit (ICT-MobileSummit'09)*, 2009.

[CBG16] Ray Chen, Fenye Bao, and Jia Guo. "Trust-Based Service Management for Social Internet of Things Systems." *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 6, 2016, pp. 684–696.

[CC13] Michael J. Covington and Rush Carskadden. "Threat Implications of the Internet of Things." *Cyber Conflict (CyCon), 2013 5th International Conference On*. IEEE, 2013, pp. 1–12.

[Cer99] Certicom. "SEC 2: Recommended Elliptic Curve Domain Parameters," 1999. URL: `https://www.secg.org/SEC2-Ver-1.0.pdf` (visited on 04/08/2017).

[Che+11] Dong Chen et al. "TRM-IoT: A Trust Management Model Based on Fuzzy Reputation for Internet of Things." *Computer Science and Information Systems*, vol. 8, no. 4, 2011, pp. 1207–1228.

[Civ] Civic. *Civic Secure Identity Ecosystem - Decentralized Identity & Reusable KYC*. URL: `https://www.civic.com/` (visited on 08/31/2018).

[Cloa] CloudPlugs. *Industrial Internet of Things*. URL: `https://cloudplugs.com/industrial-internet-of-things/` (visited on 08/25/2018).

[Clob] CloudPlugs. *Internet Of Things Platform, Industrial IoT, Industry 4.0, Public Cloud*. URL: `https://cloudplugs.com/` (visited on 08/24/2018).

[CM12] Moumena A. Chaqfeh and Nader Mohamed. "Challenges in Middleware Solutions for the Internet of Things." *Collaboration Technologies and Systems (CTS), 2012 International Conference On*. IEEE, 2012, pp. 21–26.

[Cob15] Pamela Cobb. *German Steel Mill Meltdown: Rising Stakes in the Internet of Things*. 2015. URL: https://securityintelligence.com/german-steel-mill-meltdown-rising-stakes-in-the-internet-of-things/ (visited on 06/23/2018).

[Con] ConsenSys. *Harness the Power of Ethereum*. URL: https://new.consensys.net/ (visited on 08/31/2018).

[Con+12] D. Conzon et al. "The VIRTUS Middleware: An XMPP Based Architecture for Secure IoT Communications." *2012 21st International Conference on Computer Communications and Networks (ICCCN)*. 2012, pp. 1–6.

[Cor16] Corero. *The Mirai Botnet: All About the Latest Malware DDoS Attack Type | Corero*. 2016. URL: https://www.corero.com/resources/ddos-attack-types/mirai-botnet-ddos-attack.html (visited on 06/23/2018).

[CPS14] Yuan Cheng, Jaehong Park, and Ravi Sandhu. "Attribute-Aware Relationship-Based Access Control for Online Social Networks." *Proceedings of the IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2014, pp. 292–306.

[CS05] Scott Cantor and T. Scavo. "Shibboleth Architecture." *Protocols and Profiles*, vol. 10, 2005, p. 16.

[CS15] O. Can and O. K. Sahingoz. "A Survey of Intrusion Detection Systems in Wireless Sensor Networks." *2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO)*. 2015, pp. 1–6.

[CSP17] CSPAN. *Senate Banking Committee Hearing on Equifax Data Breach*. 2017. URL: https://www.c-span.org/video/?434469-1/equifax-ceo-testifies-senate-banking-panel (visited on 06/28/2018).

[DAI03] DAIDALOS. *Daidalos*. 2003. URL: https://www.tssg.org/projects/daidalos/ (visited on 06/28/2018).

[Dan+17] Thien-Binh Dang et al. "On Evaluating IoTivity Cloud Platform." International Conference on Computational Science and Its Applications. Lecture Notes in Computer Science. Springer, Cham, 2017, pp. 137–147.

[Dat+15] Soumya Kanti Datta et al. "oneM2M Architecture Based User Centric IoT Application Development." *Future Internet of Things and Cloud (FiCloud), 2015 3rd International Conference On*. IEEE, 2015, pp. 100–107.

[Dat+16] Soumya Kanti Datta et al. "oneM2M Architecture Based IoT Framework for Mobile Crowd Sensing in Smart Cities." *Proceedings of the 2016 European Conference on Networks and Communications (EuCNC)*. Athens, Greece: IEEE, 2016, pp. 168–173.

[Dat13] DataArt. *DeviceHive - Open Source IoT Data Platform with the Wide Range of Integration Options*. 2013. URL: https://devicehive.com/ (visited on 08/10/2018).

[DBP96] Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. "RIPEMD-160: A Strengthened Version of RIPEMD." *Proceedings of the International Workshop on Fast Software Encryption*. Springer, 1996, pp. 71–82.

[dCru+18] Mauro AA da Cruz et al. "A Reference Model for Internet of Things Middleware." *IEEE Internet of Things Journal*, vol. 5, 2018, pp. 871–883.

[De +08] Luciana Moreira Sá De Souza et al. "Socrades: A Web Service Based Shop Floor Integration Infrastructure." *The Internet of Things*. Springer, 2008, pp. 50–67.

[Dim05] Tassos Dimitriou. "A Lightweight RFID Protocol to Protect against Traceability and Cloning Attacks." *Security and Privacy for Emerging Areas in Communications Networks, 2005.* IEEE, 2005, pp. 59–66.

[Din+17] Tien Tuan Anh Dinh et al. "Blockbench: A Framework for Analyzing Private Blockchains." *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 2017, pp. 1085–1100.

[DO 07] IVAR JØRSTAD DO VAN THANH. "The Ambiguity of Identity." *Identity Management*, 2007, p. 3.

[Dor+17] Ali Dorri et al. "Blockchain for IoT Security and Privacy: The Case Study of a Smart Home." *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2017, pp. 618–623.

[Dou02] Blake Dournaee. *XMl Security*. New York, NY, USA: McGraw-Hill, Inc., 2002. ISBN: 978-0-07-219399-2.

[DPB13] Flávia C. Delicato, Paulo F. Pires, and Thais Batista. *Middleware Solutions for the Internet of Things*. Springer, 2013.

[DYW11] Chao Ding, L. J. Yang, and Meng Wu. "Security Architecture and Key Technologies for IoT/CPS." *ZTE technology journal*, vol. 17, no. 1, 2011, pp. 11–16.

[EE12] David Evans and David M. Eyers. "Efficient Data Tagging for Managing Privacy in the Internet of Things." *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications (GreenCom)*. IEEE, 2012, pp. 244–248.

[Ele14] Carlos Elena-Lenz. *Internet of Things: Six Key Characteristics*. 2014. URL: `https://designmind.frogdesign.com/2014/08/internet-things-six-key-characteristics/` (visited on 11/04/2015).

[ERA09] Markus Eisenhauer, Peter Rosengren, and Pablo Antolin. "A Development Platform for Integrating Wireless Devices and Sensors into Ambient Intelligence Systems." *Proceedings of the 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*. 2009, pp. 1–3.

[ERM18] Clément Elbaz, Louis Rilling, and Christine Morin. "Reactive and Adaptive Security Monitoring in Cloud Computing." *2018 IEEE 3rd International Workshops on Foundations and Applications of Self* Systems (FAS* W)*. IEEE, 2018, pp. 5–7.

[ESC16] Mahmoud Elkhodr, Seyed Shahrestani, and Hon Cheung. "A Middleware for the Internet of Things." *arXiv preprint arXiv:1604.04823*, 2016.

[Eve] Evernym. *Self-Sovereign Identity*. URL: `https://www.evernym.com/` (visited on 08/31/2018).

[EVR] EVRYTHNG. *EVRYTHNG IoT Smart Products Platform*. URL: `https://evrythng.com/` (visited on 08/25/2018).

[Fab+19] Benedict Faber et al. "BPDIMS: A Blockchain-Based Personal Data and Identity Management System." *Proceedings of the 52nd Hawaii International Conference on System Sciences*. 2019.

[Far+15] Ivan Farris et al. "Taking the SIoT down from the Cloud: Integrating the Social Internet of Things in the INPUT Architecture." *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. 2015, pp. 35–39.

[Fer+14] Hiro Gabriel Cerqueira Ferreira et al. "Proposal of a Secure, Deployable and Transparent Middleware for Internet of Things." *Proceedings of the 9th Iberian Conference on Information Systems and Technologies (CISTI)*. 2014, pp. 1–4.

[Fer15] Ghofrane Fersi. "Middleware for Internet of Things: A Study." *Proceedings of the 2015 International Conference on Distributed Computing in Sensor Systems*. Fortaleza, Brazil: IEEE, 2015, pp. 230–235.

[FG12] Kai Fischer and Jürgen Geßner. "Security Architecture Elements for IoT Enabled Automation Networks." *Proceedings of the 2012 IEEE 17th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2012, pp. 1–8.

[FHN07] Samer Fayssal, Salim Hariri, and Youssif Al-Nashif. "Anomaly-Based Behavior Analysis of Wireless Network Security." *Proceedings of the Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services*. IEEE, 2007, pp. 1–8.

[FLF12] Christian Fuhrhop, John Lyle, and Shamal Faily. "The Webinos Project." *Proceedings of the 21st International Conference on World Wide Web*. ACM, 2012, pp. 259–262.

[Fon11] Philip WL Fong. "Relationship-Based Access Control: Protection Model and Policy Language." *Proceedings of the First ACM Conference on Data and Application Security and Privacy*. ACM, 2011, pp. 191–202.

[For96] Peter Forrest. "The Identity of Indiscernibles," 1996. URL: `https://plato.stanford.edu/archives/win2016/entries/identity-indiscernible/` (visited on 06/28/2018).

[Fre14] Paul Fremantle. "A Reference Architecture for the Internet of Things." *WSO2 White paper*, 2014.

[Fre15] Paul Fremantle. "A Reference Architecture for the Internet of Things," 2015.

[FS17] Paul Fremantle and Philip Scott. "A Survey of Secure Middleware for the Internet of Things." *PeerJ Computer Science*, vol. 3, 2017, e114.

[Fuh13] Christian Fuhrhop. "D3.7: Final Webinos Specification," 2013, p. 1046.

[Fuq+15] Ala Al-Fuqaha et al. "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications." *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, 2015, pp. 2347–2376.

[FVY14a] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. "A Decentralized Public Key Infrastructure with Identity Retention." *IACR Cryptology ePrint Archive*, vol. 2014, 2014, p. 803.

[FVY14b] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. "CertCoin: A NameCoin Based Decentralized Authentication System 6.857 Class Project," 2014. URL: `http://courses.csail.mit.edu/6.857/2014/files/19-fromknecht-velicann-yakoubov-certcoin.pdf` (visited on 02/20/2017).

[Gao+18] Zhimin Gao et al. "Blockchain-Based Identity Management with Mobile Device." *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems - CryBlock'18*. Munich, Germany: ACM Press, 2018, pp. 66–70.

[Gar] Gartner. *Gartner: Fueling the Future of Business*. URL: https://www.gartner.com/en (visited on 09/02/2018).

[Gar17] Gartner. *Gartner Says 8.4 Billion Connected*. 2017. URL: https://www.gartner.com/newsroom/id/3598917 (visited on 06/23/2018).

[GBM18] Arthur Gatouillat, Youakim Badr, and Bertrand Massot. "Smart and Safe Self-Adaption of Connected Devices Based on Discrete Controllers." *IET Software*, vol. 13, no. 1, 2018, pp. 49–59.

[Ger+16] Arthur Gervais et al. "On the Security and Performance of Proof of Work Blockchains." *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 3–16.

[GG15] Larry Greenemeier and Larry Greenemeier. *Recall Shows That a Hack Attack on Car Controls Is a Credible Threat*. 2015. URL: https://www.scientificamerican.com/article/recall-shows-that-a-hack-attack-on-car-controls-is-a-credible-threat/ (visited on 06/23/2018).

[Gia13] Raffaele Giaffreda. "iCore: A Cognitive Management Framework for the Internet of Things." *The Future Internet Assembly*. Springer, 2013, pp. 350–352.

[Gli11] Alex Glikson. "Fi-Ware: Core Platform for Future Internet Applications." *Proceedings of the 4th Annual International Conference on Systems and Storage*. 2011.

[GMA16] Roberto Girau, Salvatore Martis, and Luigi Atzori. "Neighbor Discovery Algorithms for Friendship Establishment in the Social Internet of Things." *Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. IEEE, 2016, pp. 165–170.

[GMO16] Irene Giacomelli, Jesper Madsen, and Claudio Orlandi. "ZKBoo: Faster Zero-Knowledge for Boolean Circuits." *USENIX Security Symposium*. 2016, pp. 1069–1083.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. "The Knowledge Complexity of Interactive Proof Systems." *SIAM Journal on computing*, vol. 18, no. 1, 1989, pp. 186–208.

[Góm+14] Aitor Gómez-Goiri et al. "Otsopack: Lightweight Semantic Framework for Interoperable Ambient Intelligence Applications." *Computers in Human Behavior*, vol. 30, 2014, pp. 460–467.

[Gooa] Google. *Cloud IoT Core*. URL: https://cloud.google.com/iot-core/ (visited on 08/22/2018).

[Goob] Google. *Cloud IoT Edge - Extending Google Cloud's AI & ML*. URL: https://cloud.google.com/iot-edge/ (visited on 08/22/2018).

[Gooc] Google. *Google Cloud IoT - Fully Managed IoT Services*. URL: https://cloud.google.com/solutions/iot/ (visited on 08/22/2018).

[GPR12] Sergio Gusmeroli, Salvatore Piccione, and Domenico Rotondi. "IoT@Work Automation Middleware System Design and Architecture." *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA 2012)*. 2012, pp. 1–8.

[GPR13] 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*, vol. 58, 2013, pp. 1189–1205.

[Gre15] Larry Greenemeier. "Recall Shows That a Hack Attack on Car Controls Is a Credible Threat." *Scientific American*, 2015.

[Gui08] Christopher Guindon (Webdev). *Eclipse Higgins*. 2008. URL: `http://projects.eclipse.org/projects/technology.higgins` (visited on 06/28/2018).

[Hal17] Harry Halpin. "NEXTLEAP: Decentralizing Identity with Privacy for Secure Messaging." Proceedings of the 12th International Conference on Availability, Reliability and Security. ACM, 2017, p. 92.

[Har12] Dick Hardt. "The OAuth 2.0 Authorization Framework," 2012.

[Has+13] Omar Hasan et al. "A Decentralized Privacy Preserving Reputation Protocol for the Malicious Adversarial Model." *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 6, 2013, pp. 949–962.

[Hay+09] Simon S. Haykin et al. *Neural Networks and Learning Machines*. Vol. 3. Pearson Upper Saddle River, NJ, USA: 2009.

[HBB12] Omar Hasan, Lionel Brunie, and Elisa Bertino. "Preserving Privacy of Feedback Providers in Decentralized Reputation Systems." *Computers & Security*, vol. 31, no. 7, 2012, pp. 816–826.

[HBF17] Dina Hussein, Emmanuel Bertin, and Vincent Frey. "A Community-Driven Access Control Approach in Distributed IoT Environments." *IEEE Communications Magazine*, vol. 55, no. 3, 2017, pp. 146–153.

[HE] Tony Hansen and Donald Eastlake. *US Secure Hash Algorithms (SHA and SHA-Based HMAC and HKDF)*. URL: `https://tools.ietf.org/html/rfc6234` (visited on 05/16/2019).

[Her+13] José L. Hernández-Ramos et al. "Distributed Capability-Based Access Control for the Internet of Things." *Journal of Internet Services and Information Security (JISIS)*, vol. 3, no. 3/4, 2013, pp. 1–16.

[HFH15] Md Mahmud Hossain, Maziar Fotouhi, and Ragib Hasan. "Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things." *Proceedings of the 2015 IEEE World Congress on Services*. IEEE, 2015, pp. 21–28.

[HG10] Antonio Garrote Hernández and María N. Moreno García. "A Formal Definition of RESTful Semantic Web Services." *Proceedings of the First International Workshop on RESTful Design*. ACM, 2010, pp. 39–45.

[HM05] John Hughes and Eve Maler. "Security Assertion Markup Language (Saml) v2. 0 Technical Overview." *OASIS SSTC Working Draft sstc-saml-tech-overview-2.0-draft-08*, 2005, pp. 29–38. URL: `https://www.oasis-open.org/committees/download.php/14361/sstc-saml-tech-overview-2.0-draft-08.pdf` (visited on 04/07/2017).

[Hod02] Damian Hodgson. "Know Your Customer: Marketing, Governmentality and the New Consumer of Financial Services." *Management Decision*, vol. 40, no. 4, 2002, pp. 318–328.

[HP +13] H.-P Huth et al. "IoT@Work - D1.3 – Final Framework Architecture Specification," 2013. DOI: `10.13140/rg.2.2.35022.97602`.

[HP16] Thomas Hardjono and Alex Sandy Pentland. *Verifiable Anonymous Identities and Access Control in Permissioned Blockchains*. 2016.

[Hua+12] Xin Huang et al. "User Interactive Internet of Things Privacy Preserved Access Control." *Proceedings of the 2012 International Conference for Internet Technology And Secured Transactions*. IEEE, 2012, pp. 597–602.

[IBM] IBM. *IoT Platform - IBM Watson IoT*. URL: https://www.ibm.com/internet-of-things/spotlight/watson-iot-platform (visited on 08/27/2018).

[IBM04] IBM. *Introduction to Service Data Objects*. 2004. URL: http://www.ibm.com/developerworks/library/j-sdo/index.html (visited on 09/02/2018).

[ID2] ID2020. *An Alliance Commited to Improving Lives Through Digital Identity*. URL: https://id2020.org/ (visited on 08/31/2018).

[IF18] Mike Isaac and Sheera Frenkel. "Facebook Security Breach Exposes Accounts of 50 Million Users." *The New York Times. Technology*, 2018. ISSN: 0362-4331. URL: https://www.nytimes.com/2018/09/28/technology/facebook-hack-data-breach.html (visited on 01/21/2019).

[Isa+12] Mohd Anuar Mat Isa et al. "A Lightweight and Secure TFTP Protocol for Smart Environment." *Proceedings of the 2012 IEEE Symposium on Computer Applications and Industrial Electronics (ISCAIE)*. IEEE, 2012, pp. 302–306.

[ITU09] ITU-T. *ITU-T Recommendation Y.2720 NGN Identity Management Framework, Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks*. 2009. URL: http://www.itu.int/ITU-T/recommendations/rec.aspx?id=9574&lang=en (visited on 04/07/2017).

[ITU12a] ITU-T. *ITU-T Recommendation Y.2060 Overview of the Internet of Things, Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks*. 2012. URL: http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11559 (visited on 10/26/2015).

[ITU12b] ITU-T. *ITU-T Recommendation Y.2069 Terms and Definitions for the Internet of Things, Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks*. 2012. URL: http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11700 (visited on 10/26/2015).

[JKO13] Marek Jawurek, Florian Kerschbaum, and Claudio Orlandi. "Zero-Knowledge Using Garbled Circuits: How to Prove Non-Algebraic Statements Efficiently." *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. ACM, 2013, pp. 955–966.

[JNL11] Paria Jokar, Hasen Nicanfar, and Victor CM Leung. "Specification-Based Intrusion Detection for Home Area Networks in Smart Grids." *Proceedings of the 2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2011, pp. 208–213.

[Jol] Jolocom. *Decentralized Infrastructure for Self-Sovereign Identity*. URL: http://jolocom.io/ (visited on 08/31/2018).

[JP05] Audun Jøsang and Simon Pope. "User Centric Identity Management." *AusCERT Asia Pacific Information Technology Security Conference*. Citeseer, 2005, p. 77.

[JW00] Prasad Jogalekar and Murray Woodside. "Evaluating the Scalability of Distributed Systems." *IEEE Transactions on parallel and distributed systems*, vol. 11, no. 6, 2000, pp. 589–603.

[JZS07] Audun Jøsang, Muhammed Al Zomai, and Suriadi Suriadi. "Usability and Privacy in Identity Management Architectures." *Proceedings of the Fifth Australasian Symposium on ACSW Frontiers*. Australian Computer Society, Inc., 2007, pp. 143–152.

[Kaa14] KaaIoT. *Kaa Open-Source IoT Platform*. 2014. URL: https://kaaproject.org/ (visited on 08/14/2018).

[Kan09] Kantara. *Kantara Initiative*. 2009. URL: https://kantarainitiative.org/ (visited on 06/28/2018).

[KL14] Jaeho Kim and Jang-Won Lee. "OpenIoT: An Open Service Framework for the Internet of Things." *Proceedings of the 2014 IEEE World Forum on Internet of Things (WF-IoT)*. IEEE, 2014, pp. 89–93.

[Kob91] Neal Koblitz. "CM-Curves with Good Cryptographic Properties." *Proceedings of the Annual International Cryptology Conference*. Springer, 1991, pp. 279–287.

[KPC16] Nesrine Khernane, Maria Potop-Butucaru, and Claude Chaudet. "BANZKP: A Secure Authentication Scheme Using Zero Knowledge Proof for WBANs." *Proceedings of the 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 2016, pp. 307–315.

[Lee18] J. Lee. "BIDaaS: Blockchain Based ID As a Service." *IEEE Access*, vol. 6, 2018, pp. 2274–2278.

[Lei+16] Benjamin Leiding et al. "Authcoin: Validation and Authentication in Decentralized Networks." *arXiv preprint arXiv:1609.04955*, 2016.

[Lev+14] Tihamer Levendovszky et al. "Distributed Real-Time Managed Systems: A Model-Driven Distributed Secure Information Architecture Platform for Managed Embedded Systems." *IEEE software*, vol. 31, 2014, pp. 62–69.

[Lib01] Liberty. *The Liberty Alliance Project*. 2001. URL: http://projectliberty.org/ (visited on 06/28/2018).

[Liu+17] Yuan Liu et al. "An Identity Management System Based on Blockchain." *Proceedings of the 2017 15th Annual Conference on Privacy, Security and Trust (PST)*, 2017, pp. 44–53.

[LM14] Xiang Li and Sangman Moh. "Middleware Systems for Wireless Sensor Networks: A Comparative Survey." *Contemporary Engineering Sciences*, vol. 7, no. 13, 2014, pp. 649–660.

[Losa] Losant. *Losant Security Overview Documentation*. URL: https://docs.losant.com/security/ (visited on 08/25/2018).

[Losb] Losant. *The Losant Enterprise IoT Platform*. URL: https://www.losant.com (visited on 08/25/2018).

[Lyl+12] John Lyle et al. "On the Design and Development of Webinos: A Distributed Mobile Application Middleware." *Proceedings of the IFIP International Conference on Distributed Applications and Interoperable Systems*. Springer, 2012, pp. 140–147.

[Lyl11] John Lyle. "Webinos D3.5: Webinos Phase 1 Security Framework," 2011, p. 125.

[LYL14] Chi Harold Liu, Bo Yang, and Tiancheng Liu. "Efficient Naming, Addressing and Profile Services in Internet-of-Things Sensory Environments." *Ad Hoc Networks*, vol. 18, 2014, pp. 85–101.

[Mah+15] Rwan Mahmoud et al. "Internet of Things (IoT) Security: Current Status, Challenges and Prospective Measures." *Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*. IEEE, 2015, pp. 336–341.

[Mat+09] Vashek Matyáš et al. *The Future of Identity in the Information Society*. Vol. 298. Springer, 2009.

[McC05] Bill McCarty. *Selinux: Nsa's Open Source Security Enhanced Linux*. Vol. 238. O'Reilly, 2005.

[MGZ14] Limin Ma, Yu Ge, and Yuesheng Zhu. "TinyZKP: A Lightweight Authentication Scheme Based on Zero-Knowledge Proof for Wireless Body Area Networks." *Wireless personal communications*, vol. 77, no. 2, 2014, pp. 1077–1090.

[Mic] Microsoft Azure. *Azure Internet of Things – IoT for Every Business*. URL: `https://azure.microsoft.com/en-us/overview/iot/` (visited on 08/23/2018).

[MMR17] Damiano Di Francesco Maesa, Paolo Mori, and Laura Ricci. "Blockchain Based Access Control." *Distributed Applications and Interoperable Systems*. Proceedings of the IFIP International Conference on Distributed Applications and Interoperable Systems. Springer, Cham, 2017, pp. 206–220.

[Mon09] Douglas C. Montgomery. *Statistical Quality Control*. Vol. 7. Wiley New York, 2009.

[Mou+15] Hayam Mousa et al. "Trust Management and Reputation Systems in Mobile Participatory Sensing Applications: A Survey." *Computer Networks*, vol. 90, 2015, pp. 49–73.

[MR08] Eve Maler and Drummond Reed. "The Venn of Identity: Options and Issues in Federated Identity Management." *IEEE Security & Privacy*, vol. 6, 2008.

[MW11] Pratyusa K. Manadhata and Jeannette M. Wing. "An Attack Surface Metric." *IEEE Transactions on Software Engineering*, vol. 37, no. 3, 2011, pp. 371–386.

[Nak08] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. URL: `http://www.cryptovest.co.uk/resources/Bitcoin%20paper%20Original.pdf` (visited on 06/22/2016).

[Nam14] Namecoin. *Namecoin*. 2014. URL: `https://namecoin.org/` (visited on 06/28/2018).

[Nas+09] Pascal Bou Nassar et al. "Risk Management and Security in Service-Based Architectures." *Proceedings of the 2009 International Conference on Advances in Computational Tools for Engineering Applications*. IEEE, 2009, pp. 214–218.

[Nei+14] Ricardo Neisse et al. "A Model-Based Security Toolkit for the Internet of Things." *Proceedings of the 2014 Ninth International Conference on Availability, Reliability and Security (ARES)*. IEEE, 2014, pp. 78–87.

[NGA14] Michele Nitti, Roberto Girau, and Luigi Atzori. "Trustworthiness Management in the Social Internet of Things." *IEEE Transactions on knowledge and data engineering*, vol. 26, no. 5, 2014, pp. 1253–1266.

[Ngu+17] Anne H Ngu et al. "IoT Middleware: A Survey on Issues and Enabling Technologies." *IEEE Internet of Things Journal*, vol. 4, 2017, pp. 1–20.

[Nit+12] Michele Nitti et al. "A Subjective Model for Trustworthiness Evaluation in the Social Internet of Things." *Proceedings of the 2012 IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2012, pp. 18–23.

[Nit+17] Michele Nitti et al. "Trustworthiness Management in the IoT: The Importance of the Feedback." *Proceedings of the 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*. 2017, pp. 325–327.

[Noe15] Shen Noether. "Ring SIgnature Confidential Transactions for Monero." *IACR Cryptology ePrint Archive*, vol. 2015, 2015, p. 1098.

[OAA16] Aafaf Ouaddah, Anas Abou Elkalam, and Abdellah Ait Ouahman. "FairAccess: A New Blockchain-Based Access Control Framework for the Internet of Things." *Security and Communication Networks*, vol. 9, no. 18, 2016, pp. 5943–5964.

[OASa] OASIS. *Advancing Open Standards for the Information Society*. URL: `https://www.oasis-open.org/` (visited on 09/02/2018).

[OASb] OASIS. *Service Component Architecture (SCA)*. URL: `http://www.oasis-opencsa.org/sca` (visited on 09/02/2018).

[OHL17] Aissam Outchakoucht, ES-SAMAALI Hamza, and Jean Philippe Leroy. "Dynamic Access Control Policy Based on Blockchain and Machine Learning for the Internet of Things." *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 7, 2017, pp. 417–424.

[P+09] Ganapathi Padmavathi, D. Shanmugapriya, et al. "A Survey of Attacks, Security Mechanisms and Challenges in Wireless Sensor Networks." *arXiv preprint arXiv:0909.0576*, 2009.

[Par+10] David Parlanti et al. "A Scalable Grid and Service-Oriented Middleware for Distributed Heterogeneous Data and System Integration in Context-Awareness-Oriented Domains." *The Internet of Things*. Springer, 2010, pp. 109–118.

[Par+13] Bryan Parno et al. "Pinocchio: Nearly Practical Verifiable Computation." *Proceedings of the 2013 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2013, pp. 238–252.

[Pér+11] Alejandro Pérez et al. "Formal Description of the SWIFT Identity Management Framework." *Future Generation Computer Systems*, vol. 27, no. 8, 2011, pp. 1113–1123.

[PH16] Jesus Pacheco and Salim Hariri. "IoT Security Framework for Smart Cyber Infrastructures." *Proceedings of the 2016 IEEE International Workshops on Foundations and Applications of Self* Systems*. IEEE, 2016, pp. 242–247.

[PIC07] PICOS. *Privacy and Identity Management for Community Services*. 2007. URL: `http://www.picos-project.eu/` (visited on 06/28/2018).

[PRI11] PRIMELife. *Privacy and Identity Management in Europe for Life*. 2011. URL: `http://primelife.ercim.eu/` (visited on 06/28/2018).

[Raz+16] Mohammad Abdur Razzaque et al. "Middleware for Internet of Things: A Survey." *IEEE Internet of Things Journal*, vol. 3, no. 1, 2016, pp. 70–95.

[Ren+11] Yi Ren et al. "Security in Mobile Wireless Sensor Networks a Survey." *Journal of Communications*, vol. 6, no. 2, 2011, pp. 128–142.

[RLN11] Rodrigo Roman, Javier Lopez, and Pablo Najera. "A Cross-Layer Approach for Integrating Security Mechanisms in Sensor Networks Architectures." *Wireless Communications and Mobile Computing*, vol. 11, no. 2, 2011, pp. 267–276.

[Row+11] Anthony Rowe et al. "Sensor Andrew: Large-Scale Campus-Wide Sensing and Actuation." *IBM Journal of Research and Development*, vol. 55, no. 1.2, 2011, pp. 6–1.

[RR06] David Recordon and Drummond Reed. "OpenID 2.0: A Platform for User-Centric Identity Management." *Proceedings of the Second ACM Workshop on Digital Identity Management*. ACM, 2006, pp. 11–16.

[RSA17] RSAConference. *MEDJACK.3: New Research on Attacks on Hospital Medical Devices | USA 2017 | RSA Conference*. 2017. URL: http://www.rsaconference.com/events/us17/agenda/sessions/4518-medjack-3-new-research-on-attacks-on-hospital (visited on 06/23/2018).

[RZL13] Rodrigo Roman, Jianying Zhou, and Javier Lopez. "On the Features and Challenges of Security and Privacy in Distributed Internet of Things." *Computer Networks*, vol. 57, no. 10, 2013, pp. 2266–2279. ISSN: 1389-1286.

[Sam] Samsung. *IoT Cloud Platform — Samsung ARTIK Cloud Services*. URL: https://artik.cloud/ (visited on 08/21/2018).

[San+96] Ravi S. Sandhu et al. "Role-Based Access Control Models." *Computer*, vol. 29, no. 2, 1996, pp. 38–47.

[SAN14] SANS Institute. *Securing the Internet of Things Survey*. 2014, p. 10. URL: https://www.sans.org/reading-room/whitepapers/covert/securing-internet-things-survey-34785 (visited on 08/22/2018).

[Sar+15] Chayan Sarkar et al. "DIAT: A Scalable Distributed Architecture for IoT." *IEEE Internet of Things Journal*, vol. 2, no. 3, 2015, pp. 230–239.

[Sas+14] Eli Ben Sasson et al. "Zerocash: Decentralized Anonymous Payments from Bitcoin." *2014 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2014, pp. 459–474.

[Ser+13] Alexandru Serbanati et al. "Internet of Things Architecture." *Concept and Solutions for Privacy and Security in the Resolution Infrastructure, EU Project IoT-A, Project Report D*, vol. 4, 2013.

[SFJ09] Suriadi Suriadi, Ernest Foo, and Audun Jøsang. "A User-Centric Federated Single Sign-on System." *Journal of Network and Computer Applications*, vol. 32, no. 2, 2009, pp. 388–401.

[Sho] ShoCard. *Secure Enterprise Identity Authentication*. URL: https://shocard.com/ (visited on 08/31/2018).

[Sic+15] Sabrina Sicari et al. "Security, Privacy and Trust in Internet of Things: The Road Ahead." *Computer Networks*, vol. 76, 2015, pp. 146–164.

[Sic+16] Sabrina Sicari et al. "A Secure and Quality-Aware Prototypical Architecture for the Internet of Things." *Information Systems*, vol. 58, 2016, pp. 43–55.

[Sit15] SiteWhere. *The Open Platform for the Internet of Things*. 2015. URL: http://www.sitewhere.org/ (visited on 08/15/2018).

[Sov] Sovrin. *Identity For All*. URL: https://sovrin.org/ (visited on 08/31/2018).

[SS17] Pallavi Sethi and Smruti R. Sarangi. "Internet of Things: Architectures, Protocols, and Applications." *Journal of Electrical and Computer Engineering*, vol. 2017, 2017.

[STO08] STORK. *Secure idenTity acrOss boRders linKed 2.0*. 2008. URL: https://www.eid-stork2.eu/ (visited on 06/28/2018).

[Sub15] Ashok Subash. "IoTivity–Connecting Things in IoT." *TIZEN Development Summit*, 2015.

[Suo+12] Hui Suo et al. "Security in the Internet of Things: A Review." *Proceedings of the 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE)*. Vol. 3. 2012, pp. 648–651.

[Swe+14] Jorg Swetina et al. "Toward a Standardized Common M2M Service Layer Platform: Introduction to oneM2M." *IEEE Wireless Communications*, vol. 21, no. 3, 2014, pp. 20–26.

[Tel] Telit. *Telit: IoT Solutions Provider - Modules, IoT Platforms & IoT Connectivity.* URL: `https://www.telit.com/` (visited on 08/25/2018).

[TK16] Ondrej Tomanek and Lukas Kencl. "Security and Privacy of Using AllJoyn IoT Framework at Home and Beyond." *Proceedings of the 2016 2nd International Conference on Intelligent Green Building and Smart Grid (IGBSG).* IEEE, 2016, pp. 1–6.

[TNP13] Jenny Torres, Michele Nogueira, and Guy Pujolle. "A Survey on Identity Management for the Future Network." *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, 2013, pp. 787–802.

[Ubi] Ubidots. *Ubidots IoT Platform.* URL: `https://ubidots.com/` (visited on 08/26/2018).

[Upo] Uport. *Open Identity System for the Decentralized Web.* URL: `https://www.uport.me/` (visited on 08/31/2018).

[Val+16] Carlo Vallati et al. "Betaas: A Platform for Development and Execution of Machine-to-Machine Applications in the Internet of Things." *Wireless Personal Communications*, vol. 87, no. 3, 2016, pp. 1071–1091.

[Vas+15] Emmanouil Vasilomanolakis et al. "On the Security and Privacy of Internet of Things Architectures and Systems." *Secure Internet of Things (SIoT), 2015 International Workshop On.* IEEE, 2015, pp. 49–57.

[Vla+13] Panagiotis Vlacheas et al. "Enabling Smart Cities through a Cognitive Management Framework for the Internet of Things." *IEEE Communications Magazine*, vol. 51, no. 6, 2013, pp. 102–111.

[VRR07] Ricardo Valerdi, Adam M. Ross, and Donna H. Rhodes. "A Framework for Evolving System of Systems Engineering." *CrossTalk.* 2007.

[W3C] W3C. *World Wide Web Consortium.* URL: `https://www.w3.org/` (visited on 09/02/2018).

[W3C07] W3C. *Web Services Policy 1.5 - Framework.* 2007. URL: `https://www.w3.org/TR/2007/REC-ws-policy-20070904/` (visited on 09/02/2018).

[Wan+12] Zhu Wang et al. "Multi-Agent Control System with Information Fusion Based Comfort Model for Smart Buildings." *Applied Energy*, vol. 99, 2012, pp. 247–254.

[Win15] Wind River. "Security in the Internet of Things: Lessons from the Past for the Connected Future," 2015.

[Woo14] Gavin Wood. "Ethereum: A Secure Decentralised Generalised Transaction Ledger," 2014, p. 39.

[WSO] WSO2. *The Open Source Technology for Digital Business.* URL: `https://wso2.com/` (visited on 08/27/2018).

[WW11] Yanjiong Wang and Qiaoyan Wen. "A Privacy Enhanced Dns Scheme for the Internet of Things," 2011.

[Xiv] Xively. *IoT Platform for Connected Devices – Xively.* URL: `https://xively.com/` (visited on 08/25/2018).

[Yi+15] Shanhe Yi et al. "Fog Computing: Platform and Applications." *Proceedings of the 2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb).* IEEE, 2015, pp. 73–78.

[Yun+15] Jaeseok Yun et al. "A Device Software Platform for Consumer Electronics Based on the Internet of Things." *IEEE Transactions on Consumer Electronics*, vol. 61, no. 4, 2015, pp. 564–571.

[ZB18] Xiaoyang Zhu and Badr, Youakim. "Fog Computing Security Architecture for the Internet of Things Using Blockchain-Based Social Networks." *Proceedings of the 2018 IEEE Symposium on Blockchain*. 2018, pp. 1361–1366.

[Zhu09] Xuemei Zhu. "Practical PID Controller Implementation and the Theory Behind." *Proceedings of the 2009 Second International Conference on Intelligent Networks and Intelligent Systems*. IEEE, 2009, pp. 58–61.

[Zoo18] Zooko. *Zooko's Triangle. Wikipedia*. Page Version ID: 846796887. 2018. URL: `https://en.wikipedia.org/w/index.php?title=Zooko's_triangle&oldid=846796887` (visited on 06/28/2018).

[ZV10] Junqi Zhang and Vijay Varadharajan. "Wireless Sensor Network Key Management Survey and Taxonomy." *Journal of Network and Computer Applications*, vol. 33, no. 2, 2010, pp. 63–75.

Glossary

ACC Access Control Challenges 10, 14, 137
ACL Access Control List 30
AES Advanced Encryption Standard 24, 25
AMQP Advanced Message Queuing Protocol 28

BAN Burrows–Abadi–Needham 16, 70, 87, 88, 91, 92, 137

CA Certificate Authority 31
CoAP Constrained Application Protocol 21, 22, 29, 31, 33

DHKE Diffie-Hellman Key Exchange 24
DHT Distributed Hash Table 88, 90, 114, 124
DoS Denial of Service 2, 10–12, 38, 95
DTLS Datagram Transport Layer Security 30, 31

EU European Union 26, 47

GUI Graphical User Interface 109, 115

HTTP Hypertext Transfer Protocol 29

IDC Identity Challenges 8, 9, 13, 14, 137
IdM Identity Management 76, 90–92, 95
IdP identity provider 6, 8, 13–17, 70, 91, 92, 114
IDS Intrusion Detection Systems 42
ITU International Telecommunication Union 35

JSON JavaScript Object Notation 112, 133
JWT JSON Web Token 28

LWM2M Lightweight Machine to Machine 31

MQTT Message Queuing Telemetry Transport 21, 22, 28–34, 111

OS Operating Systems 30
OSGi Open Services Gateway initiative 23

P2P Peer to Peer 24, 30, 45, 58, 60–62, 64, 114–116, 118, 119, 126, 136, 155

QoS Quality of Service 27

REST REpresentational State Transfer 14, 24, 25, 28–34, 51, 54, 57, 110, 111, 118, 137

SCADA Supervisory Control and Data Acquisition 33
SOA Service Oriented Architecture 14–16, 20, 24–26, 35, 51, 53–57, 59, 67, 68, 110, 135
SOAP Simple Object Access Protocol 51

STOMP Streaming Text Oriented Messaging Protocol 28

TLS Transport Layer Security 22, 25, 28, 30, 31, 33, 34
TSC Triple Space Computing 24

UML Unified Modeling Language . . . . . viii, 112, 113
UPnP Universal Plug and Play 25

W3C World Wide Web Consortium 51
WSDL Web Services Description Language 51

XML Extensible Markup Language 26
XMPP eXtensible Messaging and Presence Protocol 21, 23, 24, 30

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : ZHU
(avec précision du nom de jeune fille, le cas échéant)

DATE de SOUTENANCE : 24 Juin 2019

Prénoms : Xiaoyang

TITRE :

Building A Secure Infrastructure for IoT Systems in Distributed Environments

NATURE : Doctorat

Numéro d'ordre : 2019LYSEI038

Ecole doctorale : ED512 - InfoMaths

Spécialité : Informatique

RESUME :

Le principe de l'Internet des objets (IdO) est d'interconnecter non seulement les capteurs, les appareils mobiles et les ordinateurs, mais aussi les particuliers, les maisons, les bâtiments intelligents et les villes, ainsi que les réseaux électriques et hydrauliques, les automobiles et les avions, pour n'en citer que quelques-uns. Toutefois, la réalisation de la connectivité étendue de l'IdO tout en assurant la sécurité et la confidentialité des utilisateurs reste un défi. Les systèmes IdO présentent de nombreuses caractéristiques non conventionnelles, telles que l'évolutivité, l'hétérogénéité, la mobilité et les ressources limitées, qui rendent les solutions de sécurité Internet existantes inadaptées aux systèmes basés sur IdO. En outre, l'IdO préconise des réseaux peer-to-peer où les utilisateurs, en tant que propriétaires, ont l'intention d'établir des politiques de sécurité pour contrôler leurs dispositifs ou services au lieu de s'appuyer sur des tiers centralisés. En nous concentrant sur les défis scientifiques liés aux caractéristiques non conventionnelles de l'IdO et à la sécurité centrée sur l'utilisateur, nous développons une infrastructure sécurisée de l'IdO rendue possible par la technologie de la chaîne de blocs pilotée par des réseaux peer-to-peer sans confiance. Notre infrastructure sécurisée pour l'IdO permet non seulement l'identification des individus et des collectifs (entreprises, familles, organisations), mais aussi l'identification fiable des objets de l'IdO (dispositifs, services) par leurs propriétaires en se référant à la chaîne de blocage dans les réseaux peer-to-peer sans confiance. La chaîne de blocs fournit à notre infrastructure sécurisée de l'IdO une base de données fiable, immuable et publique qui enregistre les identités individuelles et collectives, ce qui facilite la conception du protocole d'authentification simplifié de l'IdO sans dépendre des fournisseurs d'identité tiers. En outre, notre infrastructure sécurisée pour l'IdO adopte un paradigme d'IdO socialisé qui permet à toutes les entités de l'IdO (à savoir les individus, les collectifs, les choses) d'établir des relations et rend l'IdO extensible et omniprésent les réseaux où les propriétaires peuvent profiter des relations pour définir des politiques d'accès pour leurs appareils ou services. En outre, afin de protéger les opérations de notre infrastructure sécurisée pour l'IdO contre tout type de menace, nous introduisons également un mécanisme autonome de détection des menaces en complément de notre cadre de contrôle d'accès, qui peut surveiller en permanence le comportement anormal des opérations de dispositif ou de service et nourrir le résultat à notre cadre de contrôle d'accès. Enfin, nous simulons ces contributions, nous présentons des cas d'utilisation et nous effectuons des expériences et des simulations qui montrent que l'infrastructure sécurisée que nous proposons est capable d'interconnecter efficacement toutes les entités de l'IdO grâce à nos mécanismes d'authentification et d'autorisation et de détecter les menaces connues et inconnues avec des taux de détection élevés et de faibles alarmes positives erronées.

MOTS-CLÉS : Internet des objets; Blockchain; Identité; Authentification; Autorisation; Détection de la menace