République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A.MIRA-BEJAIA

Faculté des Sciences Exactes
Département Informatique

THÈSE

Présentée par

SAIDI Ahmed

Pour l'obtention du grade de

DOCTEUR EN SCIENCES

Filière : Informatique

Option : Cloud Computing

Thème

Secure Data Sharing In Cloud Computing

Soutenue le : 10/05/2022 Devant le Jury composé de :

Nom et Prénom	Grade		
Mr TARI Abdelkamel	Professeur	Univ.de Bejaia	Président
Mr NOUALI Omar	DR	CERIST, Alger	Rapporteur
Mr FARAH Zoubeyr	MCA	Univ.de Bejaia	Examinateur
Mr KHANOUCHE Mohamed Essaid	MCA	ESTIN. Bejaia	Examinateur

Année Universitaire : 2021/2022

Acknowledgements

First, I would like to thank Allah almighty for giving me the strength, knowledge, ability, and opportunity to undertake this research study.

I would like to express my sincerest gratitude and thanks to **Omar Nouali**, my thesis director, for his support, dedication, trust, and advice throughout the several years of my thesis. I am very much thankful to **Abdelouahab Amira** for his guidance and valuable advice. And the time he dedicated to improve and discuss the proposed solutions.

I would like to thank the members of the thesis committee: **Pr. Abdelkamel TARI**, **Dr. Zoubeyr FARAH** , and **Dr. Mohamed Essaid KHANOUCHE**, for accepting to be part of the committee and the dedicated time and they provided to read and assess my thesis.

I am happy to acknowledge my deepest sincere gratitude to **Pr. Mourad Debbai** and **Pr. Amr M. Youssef**, with whom I exchanged valuable ideas during my visit to Concordia University and who helped me to move forward efficiently in my work preparation.

Thank you, **Elmouatez Karbab** and **Pr. Mourad Debbabi's team**, my thesis would not have been in this shape without your support.

I would also like to thank all my colleagues at the security division, especially My friends **Djedjig** and **Krinah**, for the stimulating discussions and the fun we have had in the last years.

None of this would have been possible without the love and support of all members of my family, in particular my parents, my lovely children **Melissa**

and **Marwa**, my brothers, my brother's wife, my niece **Nelya** and my family-in-law.

Last and foremost, I want to thank my sweet wife. with whom I shared every moment of my Ph.D, and I am forever grateful for the support and patience she had during these moments. I am blessed to have her next to me.

Thanks a lot, Merci beaucoup, to everybody that contributed directly or indirectly to the realization of this thesis. Enjoy your reading.

Abstract

Today, users increasingly sharing their data through the cloud; however, the owner loses control when sensitive data is outsourced. This may result in the exposure of sensitive information without the data owners' consent. This thesis addresses the problem of access control in constrained devices relying on Attribute-based Encryption (ABE), a significant challenge that needs to be appropriately overcome. Indeed, when it comes to establishing cryptographic fine-grained access control, ABE methods have many advantages. However, these methods present many implementation difficulties owing to their complexity and substantial computational and energy overheads. To overcome this issue, we used the benefits of fog computing to create collaborative and distributed versions of ABE schemes. Our methods significantly decrease energy usage and computing overhead. The second limitation of ABE schemes is that the access policy is sent in clear text with the Cipher-text. This could allow a malicious user to compromise the legitimate user's privacy by using sensitive information. We have proposed introducing false attributes, mixed with the real attributes, to preserve the privacy of the access policy. Finally, we tackled the collaboration challenge in the same group among users to decrypt data. Indeed ABE is limited in terms of user collaboration, as they only allow assigning one access authorization to one user. To overcome this challenge, we have proposed a collaborative approach that allows users in the same group to combine their access attributes in a controlled manner to decrypt the data.

Keywords. Attribute Based Encryption, Data Sharing, Decryption Outsourcing, Fog Computing, Collaboration.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, Cloud computing offers many services like computing, storage, networking, etc, and allows users to access computer resources on-demand over the Internet. Cloud computing allows users to share their data by outsourcing them to distant servers on an as-needed basis, and provides mobility and flexibility since access is available from any location through the Internet. With the development of numerous domains and their applications such as the Internet of Things (IoT), smart cities, and driverless vehicles, Cloud computing plays more than ever a critical role in the data supply/demand equation.

On the other hand, Cloud computing operates in a centralized manner, which makes it difficult to create a global and adaptable platform especially with the huge number of heterogeneous and resource-constrained devices and the variety of applications. Additionally, the Cloud's centralized structure makes it challenging to satisfy the latency requirements of specific applications aimed to resource-constrained devices. As a solution to these issues, Fog computing has recently emerged to address these constraints. Fog computing is complementary to Cloud computing as it addresses low latency by handling local data at the network's edge while leaving coordination and global analytics to the Cloud.

Even though the fog addresses many flaws present in cloud architectures, some security

issues still exit, like the privacy of sensitive data. Indeed, when sensitive data is outsourced to the Cloud or is processed by Fog nodes, the owner loses control over it. This can lead to the disclosure of sensitive information without the permission of these data owners.

A necessary prerequisite for data protection is developing efficient security techniques for resource-constrained devices. Indeed, one of the challenges that might compromise resource-constrained devices like the internet of things (IoT) is the security and privacy of data exchanged or collected, which is often strongly related to users' lives. These concerns highlight the importance of imposing security procedures on applications aimed at resource-constrained devices. Security issues in resource-constrained environments are more challenging to resolve than current Internet security issues because of the limitations of theses devices like computer capability, memory, and energy, rendering traditional security methods wholly inapplicable in this situation. The main goal of this thesis is devoted to developing efficient approaches for securing data exchanged or collected by resource-constrained devices.

1.1 Research topic

Despite the benefits of data sharing in the Cloud or Fog computing, it is critical to provide robust solutions that: (1) Guarantee high security and privacy of outsourced data, (2) Overcome the increased computation overhead on a resource-constrained device, and (3) Achieve a fine-grained access control to the data.

To do this, we address in this thesis four major issues for secure data sharing:

- **Privacy in information sharing:** In a Cloud architecture, the users outsource their data to a third-party Cloud service provider. However, this can introduce privacy issues, especially if the data is sensitive. For example medical records are very sensitive in nature, and need to have sufficient protection when outsourced to third party cloud services.

- **Access control in data sharing:** Access control is one of the primary techniques that must be used to protect the data externalization process. In general, it refers to the procedures that ensure that only authorized users have access to data. When it comes to access control, researchers typically focus on the assignment of access rights and the monitoring of access. However, this procedure becomes more complicated when users need access to several domains with varying access permissions.
- **Resource limitations:** Some encryption methods have high complexity and may result in increased computation overhead. These methods are not efficient when used by constrained devices. Indeed, the latter have very limited resources storage, and computing capacity. These limitations make cryptography mechanisms impractical, and sometimes unusable to keep privacy in such resource-constrained contexts. These issues have motivated researchers to propose many techniques and solutions to reduce cryptography schemes overhead, such as computation outsourcing, compression, etc.
- **Collaboration:** Some encryption methods used in data sharing are very limited in terms of user collaboration as they only allow assigning one access authorization for one user. However, in some cases, the encrypted data is no granted individually, but is granted to a group of users that have the right access attributes and who must collaborate to have access to the data.

1.2 Contributions

In this thesis we first reviewed the architectures of Cloud computing and Fog computing as well as security mechanisms to implement security services such as confidentiality, authentication, privacy, etc. Then we have focused on access control services. In particular, we investigated the usage of attribute-based encryption (ABE) [1], as a strategy for enhancing fine-grained cryptographic access control in a resource-constrained devices.

Indeed, ABE allows one to many encryptions and keeps the encrypted data confidential even when the storage server is untrusted. However, ABE is relatively resourcing intensive both in the encryption and in the decryption processes. Additionally, ABE does not guarantee the privacy of the access policy because the latter is sent in clear text along with the ciphertext. Moreover, ABE is very limited in user collaboration, as it only allows assigning one access authorization to one user.

This thesis aimed to overcome these challenges by providing an efficient fine-grained access control that guarantee high security and privacy of outsourced data sharing and supports the privacy of the access policy and allows users within the same group to combine their attributes while satisfying the access policy.

The contributions of this thesis are:

Contribution 1: A Multi-Fog and Privacy-Preserving Data Sharing Scheme for Resource-Constrained Devices

We have proposed a new approach based on Ciphertext-Policy Attribute-Based Encryption (CP-ABE), which uses a parallel multi-fog architecture to reduce the bandwidth and to partially delegate data decryption to these Fog devices. It also ensures the privacy of the access policy by adding false attributes. Experiments demonstrate that our approach ensures the confidentiality of the data and the privacy of the access policy. The evaluation shows that our approach is more efficient compared to existing approaches.

Contribution 2: SHARE-ABE: Collaborative Encryption Based on Group-Oriented Attributes in Chained Multi-Fog Scheme

In this contribution, we leverage the ideas of contribution 1 by introducing a chained architecture and a collaborative attribute to permit a more efficient group-oriented data sharing. Fog nodes collaborate to partially decrypt the data using an original and efficient chained architecture. Furthermore, we introduce a new construction of a collaboration at-

tribute that allows users within the same group to combine their attributes while satisfying the access policy. Experiments and analyses of the security properties demonstrate that the proposed scheme is secure and efficient, especially for resource-constrained devices.

1.3 Outline

This thesis is organized as follows:

In Chapter 2, we give the main definitions of Cloud and Fog computing, their essential components and applications. After that, we highlight the fundamentals of security services. Then, we present the main challenges that are introduced by cloud-fog technologies and highlight the specificities of using resource-constrained devices in cloud/fog environments. Next, we give an overview on data sharing models in cloud/fog environments. Finally, we focus on Security mechanisms used to implement these security services and discuss three security mechanisms in detail: Cryptography, Access control and Cryptography access control.

In Chapter 3, we present data sharing for resource-constrained devices. ABE is one of the main methods used for these tasks. ABE offers collusion-resistant and fine-grained data access control. We start by giving a background information on the two variants of attribute-based encryption, Ciphertext-Policy Attribute-Based Encryption (CP-ABE), and Key-Policy Attribute-Based Encryption (KP-ABE). Following that, the challenges associated with adopting ABE are addressed. Further, we summarize the related work on ABE, considering the challenges of using such methods. We analyze and compare the existing work against a set of criteria and point out their limitations.

In Chapter 4, we propose our solution for the data sharing aimed at resource-constrained devices. We used Fog nodes to reduce the bandwidth and decrease the decryption cost by delegating the decryption process to the Fog nodes. We start the

chapter by describing the proposed approach. we detail our model and the security requirements. Then we describe the different phases of our algorithm. After that we give an analysis of the security of our solution. Finally we present the evaluation of our approach in addition to some possible application scenarios.

In Chapter 5, we present SHARE-ABE, a novel collaborative approach based on CP-ABE that is privacy-preserving and that uses Fog computing to outsource decryption operations by enforcing a new chained collaboration method between multiple Fogs. In addition, it enables the data owner to allow users of the same group to collaborate through the access policy on certain attributes to satisfy the access policy. This collaboration is permitted only for users in the same group. The chapter is organised as follows: First, the model and security requirements are presented. Then the different phases of our work are described. Finally the security analysis and performance evaluation are presented.

At the end, we conclude this thesis in Chapter 6 and shed some light on open issues and future directions.

Part I

State of The Art

Chapter 2

Cloud and Fog Computing: Fundamentals, Security and Challanges

2.1 Introduction

The desire to share data has expanded exponentially in response to the world's massive technological growth in various industries. Many cutting-edge applications, such as the Internet of Things and self-driving vehicles, need data storage and sharing through service platforms such as the Cloud, Fog computing, or any external server, and even in a completely dispersed fashion. As a result, safeguarding information transmission has become a significant concern, especially in data-sensitive applications where the data-sharing process is vulnerable to various attacks [2]. Indeed, in addition to the risks of data loss caused by eavesdropping, hacking, and even component compromise, data owners cannot completely trust Cloud and Fog service providers. As a result, they must use effective mechanisms to ensure data confidentiality.

In this chapter, we introduce the main definitions of Cloud computing, Fog, their essential components and the fundamentals of security services. We also present some challenges in secure data sharing.

2.2 Cloud Computing

Cloud computing is a technology that allows anyone connected to the Internet to use hardware and software on demand. It is defined by The National Institute of Standards and Technology (NIST) as a "model enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." The Cloud model emphasizes availability and consists of five essential characteristics, three delivery models, and four deployment models. Gartner [3] defined Cloud computing as "a style of computing where scalable and elastic IT capabilities are provided as a service to multiple external customers using Internet technologies [4]". Garter[3] discusses the aspects of Cloud, mainly concerning the industry. The concept focuses on the environment's technical features, for example, the possibility of scalability, elasticity, and internet-based solutions. According to Buyya et al.[5], Cloud computing is described as follows "a Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers.". This concept states that Cloud computing is a combination of current and emerging technologies, such as parallel and distributed computing.

Cloud computing is a natural progression and extension of new technology and methods. This latter is intended to solve issues caused by the technologies it evolved from and bring new features. In their works, Peter Mell and Tim Grance[6] identify five essential characteristics of Cloud computing (Figure 2.1) :

- On-demand self-service: Unilaterally and automatically, Cloud services and computing resources such as server time and network storage can be delivered as needed flexibly and straightforwardly to customers without each service provider's human

involvement.

- Broad network access: Consumers can interact with the Cloud providers and their computing tools remotely. Cloud tools and services are reachable from anywhere, at any time, and they are accessible via a variety of thin and thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).

- Rapid elasticity: Cloud computing allows for the scalability of Cloud services in response to business dynamics. When additional resources are needed, consumers can access them quickly and easily, and then scale back to previous levels when those resources are no longer required. For the consumer, the provisioning services tend to be infinite and can be bought in any quantity at any time.

- Resources pooling: A multi-tenant model is used in Cloud computing to pool resources such as processing power, storage, memory, and network bandwidth to support multiple customers. This can result in better resource utilization and availability, as well as lower operating costs.

- Measured service: The Cloud offers transparency for both service providers and customers by enabling Cloud users to monitor and manage their resource use.

2.2.1 Service Delivery Models

Generally, Cloud services are divided into three main categories: IaaS (infrastructure as a service), PaaS (platform as a service), and SaaS (software as a service).

- Infrastructure as a Service (IaaS): Infrastructure services are considered to be the base layer of Cloud computing systems [7]. In this model, infrastructure services, such as storage and other computing capabilities, are made available to consumers

through a network. The consumer can install and run operating systems and applications, which can dynamically scale up and down. An example of the Cloud IaaS provider is Amazon Web Services.

- Platform as a Service (PaaS): In this model, although the consumer creates the software using programing language, libraries, service, and tools from the provider, also deploys and manages applications and their hosting environments, consumers have no control over the underlying Cloud infrastructure, including the network, servers, operating systems, and storage. Google AppEngine and Windows Azure are examples of the Cloud PaaS providers.

- Software as a Service (SaaS): Software as a Service (SaaS) is a model for delivering software in which applications are hosted by a company and made available over the Internet through a simple interface such as a web browser. In this model, consumers have no control over the infrastructure and software configuration they use. Examples of the providers of SaaS are Oracle,IBM, Microsoft, etc.

2.2.2 Deployment Models

In the following, we present the four distinct and main existing models for implementing and accessing Cloud computing environments, depending on the organizational structure and provisioning location.

- Public Cloud: The most popular implementation model is public Clouds (also known as "external Clouds"). A third-party service provider makes services such as computation, storage, networks, virtualization, and applications accessible to the general public through the Internet. In this model, services are billed according to the resources used during the applicable period. Amazon Web Services, Windows Azure Services Platform, VMware, IBM's Blue Cloud, Google AppEngine, and Sun Cloud are examples of public Cloud providers.

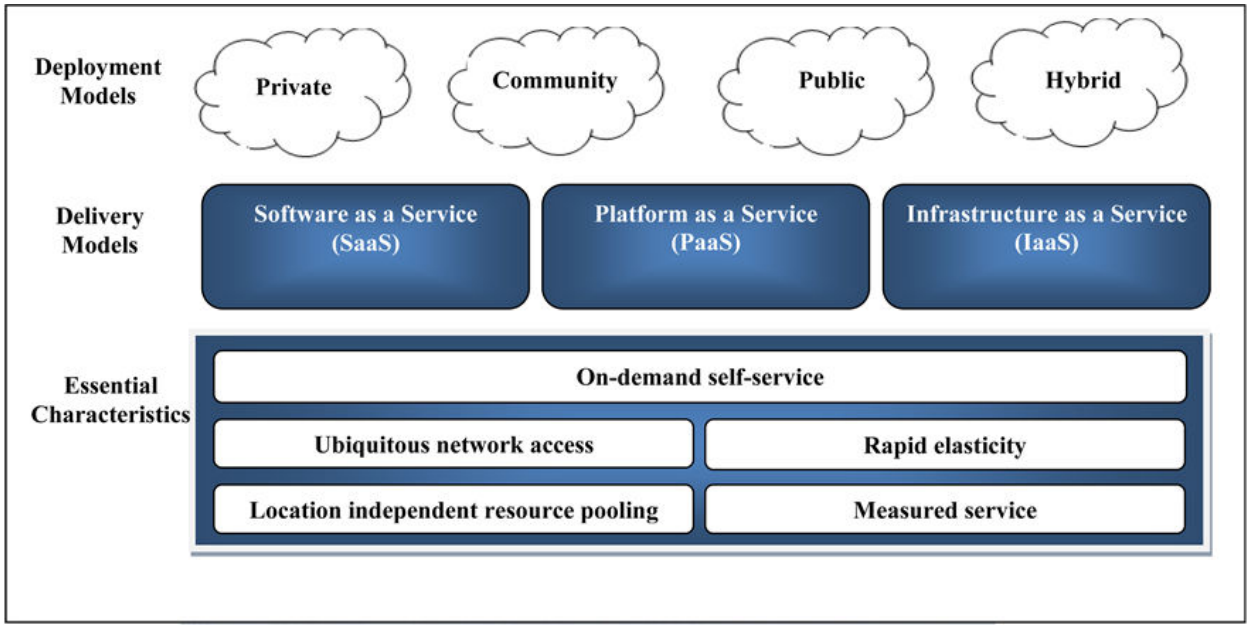

Figure 2.1: Cloud Computing Reference Model [NIST]

- Private Cloud: A private Cloud is a type of Cloud in which the technologies and platforms are specifically tailored to an organization's needs. The Cloud service provider maintains control and manages them for that purpose. A client can connect to the data and application resources in this model only if they have the necessary privileges. It removes the need for a confidence model, making it possible to introduce more flexibility. Governments and financial institutions prefer to build and use their own Clouds rather than rely on commercial ones. Amazon Web Services, VMware, Rackspace, and HP CloudStart are examples of private Cloud providers.

- Community Cloud: Organizations with similar needs (e.g., security policies and compliance considerations) share a Cloud infrastructure in a community Cloud. This model allows these organizations to pool assets and share computational resources, data storage, and other capabilities.The member organizations or a third party provider will manage the group Cloud model. As a result, it is more trustworthy than the public Cloud model and less costly than using a private Cloud.

- Hybrid Cloud: A hybrid Cloud is a Cloud that combines two or more distinct Cloud models (public, private, or community). A hybrid Cloud model is a Cloud system in which resources are handled by combining internal and external Cloud groups. The hybrid model stores confidential data internally and backups it externally in the public Cloud so that backup data is available anywhere if the system fails. The hybrid model provides the most versatility to companies by incorporating the benefits of the other models.

2.3 Fog Computing

Recently, a new concept known as Fog computing has emerged to extend Cloud services to the network's edge while maintaining interaction with the Cloud. This new paradigm is described in [8] as "a horizontal, physical or virtual resource paradigm that resides between smart end-devices and traditional Cloud or data centers."

A Fog infrastructure is primarily derived from the fundamental three-layer structure (as illustrated in Figure 2.2) and consists of IoT devices (End layer), one or more layers of Fog Nodes, and at least one Cloud Data Center (Cloud layer).

- End layer: It is the bottom layer, closest to the end-users. It consists of different IoT modules (e.g., cameras, mobile phones, smart cars, smoke detectors, . . .). These instruments, which are widely dispersed geographically, are capable of sensing events and forwarding them to the immediately upper layer of the hierarchy for processing and storage.

- Fog layer: The middle layer is made up of a set of machines that are capable of handling and storing received requests. These machines, referred to as Fog Nodes (FNs), which include access points, routers, gateways, switches, base stations, laptops, and custom Fog servers, are connected to Cloud servers and can transmit requests to data centers. These services, which are distributed between end-users and Data

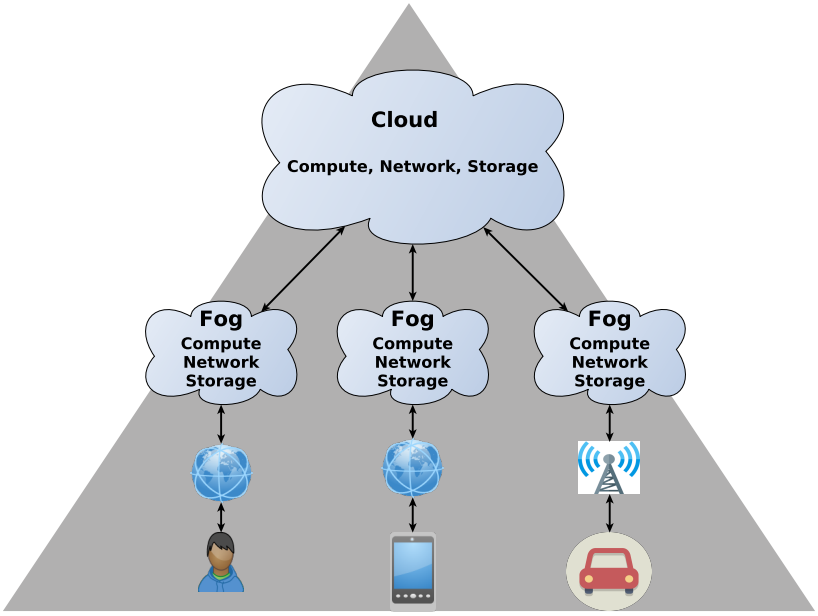

Figure 2.2: A Typical Fog Architecture[9]

Centers (DCs), can be delivered via fixed (static) devices located in a single location or via mobile devices (like smartphones, vehicles, intelligent transportation systems, drones, etc.).

- Cloud layer: The uppermost layer is made up of many servers and DCs capable of conducting complex analyses and storing massive amounts of data.

Architectural Characteristics and Advantages

Fog computing, which is regarded as the future of Cloud systems and the IoT, has a variety of characteristics and benefits, the most important of which are described below:

1. **Location awareness and low latency:** Sub-second latency-sensitive technologies such as augmented reality, virtual reality, or video streaming processes do not have to be transmitted over long paths to distant locations. Via the geo-distribution of the various Fog nodes in various locations and their proximity to end-users, Fog computing supports these aspects. Sarkar and Misra [10] established through the-

oretical modeling that Fog computing has significantly lower service latency than Cloud computing.

2. **Save bandwidth:** By performing specific computation tasks locally and sending only a subset of valuable data or those that need substantial analysis to the Cloud, Fog computing helps clear the network and accelerates specific tasks' processing.

3. **Scalability:** The number of connected devices is increasing rapidly, resulting in exponential growth in IoT data and applications. With this data being big, running it all in the Cloud is not feasible. Fog serves as a complementary paradigm that can improve device scalability.

4. **Support for mobility:** By using widely distributed Fog devices with computational and storage resources across the network, Fog computing is more suited to supporting end-user mobility than conventional centralized Cloud servers, allowing for uninterrupted service to mobile end-users.

2.4 Cloud-Fog Computing Applications

Numerous use cases are available for the Cloud and Fog computing platform. For example, Cloud and Fog computing are used in the Vehicular network, Smart-Hospitals, E-Learning, Smart-Home, etc. In this part, we present some applications that exploit the capabilities of Cloud and Fog computing.

Smart-Hospital

By using Cloud computing, healthcare organizations can improve the quality of care provided rather than maintaining their own IT, lowering or even eliminating the high expense of technical teams responsible for supporting and operating in-house infrastructures. This is a significant advantage for smaller hospitals, community care settings,

and physician offices since they may now deploy sophisticated IT infrastructures and services to support their healthcare operations without incurring significant upfront and ongoing expenses. Additionally, Cloud computing enables information exchange (internally and externally) and offers anywhere/anytime access to medical data across healthcare institutions that are engaged in the treatment process, which is critical in healthcare [11] [12].

LÃşpez et Al [13] Illustrate a typical hospital deployment scenario. Patients' physiological data and whereabouts are monitored using smart shirts in conjunction with beacons. Fog computing is a distributed computing model that utilizes many nodes. The data acquisition and processing board (DAPB) gathers, analyzes, and combines sensor data before transmitting it to the wireless transmission board (WTB). The WTB gathers data from the beacon points (BPs), merges it with data from the DAPB, and delivers it in a single packet to the management subsystem, which is situated at the LAN level. The management subsystem monitors the patients' medical parameters, locates them within the hospital, and checks if an alert has been triggered using the data from the DAPB and BPs.

Vihucalr Network

Autonomous driving requires a large amount of data and intense processing. In essence, we may outsource all computing activities to a remote cloud platform [14]. With the advent of Vehicular Cloud, many functions may now be performed directly by the local Vehicular Cloudlet. In this instance, cars serve as both collectors and processors of data. This not only saves significant amounts of transmission traffic, but also makes the autonomous driving management system more robust to Internet connectivity. Kumar et al.[15], [16] describe Carcel, an autonomous driving system aided by the Cloud. Carcel makes use of the Internet Cloud to evaluate sensor data collected from cars and roadside infrastructure in order to aid autonomous driving. Fog computing technology is also found in vehicle-based environments, such as the integration of Fog technology with ad hoc networks of

conventional vehicles (VANET) to create the Internet of Vehicles (IoV) or Fog technology for vehicles. In this latest design, vehicles are seen as intelligent, mobile, and equipped with many sensors, as well as the ability to calculate and communicate in order to gather useful traffic data. For instance, the Fog node monitors and manages local traffic flow by arranging traffic lights at each junction for the region's smart cars. At the Fog node, an intelligent traffic light control algorithm is applied (locally). The Fog node calculates traffic information such as road segment occupancy using the provided data from each vehicle and then executes the intelligent traffic light management algorithm to prevent traffic build-up by controlling each traffic signal's red and green phase percentage.

E-Learning

Today, e-learning is extensively utilized on various educational levels, including continuing education, corporate training, academic courses, etc. Cloud computing technology now is used to create e-learning systems. For example, The University of Colorado offers a service called myCUInfo' that includes all course rosters for teaching in progress, final grade submission, email, and alternative contact information, allowing teachers to administer a course from a single location. Security is essential for this kind of information, and it is often utilized as a private or virtual private Cloud service.

Smart Home

Different sensors operate in tandem to compute various operational parameters in smart buildings, including air humidity and quality, temperature, and security. These sensors collect data at various time intervals from a great distance and transmit it to a regional server. After evaluating and processing the incoming data, actuators make necessary adjustments to the building's circumstances. Several of these instances are time-sensitive and emergency-related, such as fires, healthcare systems, or stopping unauthorized people from entering. As a result, proximity to home devices is needed. This proximity may easily be achieved via Fog computing, which is situated closer to the user and is capable of delivering realtime and delay-sensitive answers, where each room and floor may have

fog nodes [17]. LÃşpez et Al [13] provides an example of a home deployment scenario using Parkinson's speech analysis. A Fog node is added to the network structure at the LAN level. As with mobile, Fog computing gathers, stores, and analyzes raw data before transferring it to the Cloud for long-term storage. Fog computing is primarily used to decrease network traffic and latency. The authors in [18] present a scenario for a home deployment, in which data from the patient- and environmental-sensing devices are utilized to identify when a patient falls and to warn caregivers of gas leaks and fires.

2.5 Fundamentals of Security Services

Security is the first consideration while migrating to the Cloud. On their personal computers, users will store a great deal of sensitive and protected data. When consumers use Cloud computing, data is transferred from their computers to the Cloud. Therefore, the Cloud should be adequately secured to protect sensitive data. Confidentiality, integrity, availability, and privacy are all issues when it comes to Cloud security. The security issues are as follow :

- **Confidentiality:** The term "confidentiality" refers to the practice of preventing unauthorized access to protected information. Users will always be concerned about confidentiality while their data is being transferred to the Cloud. Confidentiality is linked with intellectual property rights, covert channels, network traffic, data storage encryption techniques, and inference procedures in the Cloud.

- **Integrity:** The process of preserving the consistency and correctness of data is referred to as integrity. The Cloud provider should take precautions to prevent unauthorized modifications to the stored data.

- **Availability:** This ensures that data should be accessible at all times and that approved parties should have access to it if required. DDoS attacks are an example of

a threat to availability since they aim to disrupt networks, services, and applications.

- **Privacy:**Since all Cloud users' data is housed in Cloud data centers, many privacy concerns may occur. These privacy concerns include loss of control, incorrect storage, access control, and data border [19].

2.6 Cloud-Fog Computing Challenges

Cloud or Fog computing offers several benefits, including simplicity of deployment, accessibility, scalability, dependability, fault tolerance, shared resources, expanded storage capacity, and cost savings. While Cloud or Fog computing offers many benefits, it also introduces a slew of security concerns and breaches for both cloud service providers and users [20]. Providers of Cloud computing services need to address the typical security issues associated with conventional communication networks. Simultaneously, they should address other problems that are inherent in the Cloud computing paradigm. This section divides the major Cloud-Fog security problems into three categories: conventional security challenges, emerging Cloud security challenges, and emerging Fog security challenges.

2.6.1 Traditional Security Challenges

Although the security issues in conventional communication systems also apply to Cloud and fog computing, the usage of cloud and fog computing provides additional attack vectors that make assaults either conceivable or just simpler to carry out. The use of a cloud-based and fog-based system may force organisations to make many changes to the Authentication and authorization applications for businesses. In addition, Forensics operations may become much more difficult because investigators cannot physically inspect system hardware. Since Cloud and fog services affect a more significant number

of customers than in the traditional model, their availability is a major challenge.

2.6.2 Emerging Security Challenges in Cloud Computing

Because users use Cloud services and keep their data on the provider's infrastructure, privacy and confidentiality are the most pressing security concerns. Besides the owners, end-users want to know where their data is stored and who has authority over it. As well, they want assurances that even Cloud providers would not illegally access and exploit sensitive information. This section covers other significant security concerns associated with Cloud computing, including the following [21]:

- **Resource Location:** End-users use Cloud-based services without being aware of the Cloud provider's resources' actual location, which may be situated in different legislative domains. This is a potential issue when conflicts can arise and are not always within the Cloud provider's control. The policies of the Cloud service providers and the laws of the countries where the providers are located influence the data they keep. When users use such services, they must agree to the "Terms of Service," which provide providers the ability to share user information under laws and law enforcement demands

- **Multi-Tenancy issue:** This problem presents a challenge to solve for security administrators to prevent other users running processes on the same physical servers from accessing illegally user's data. This is not a brand-new problem, given the current state of affairs with web hosting providers. Cloud computing is becoming more and more widely used, and as a result, more and more valuable information is being kept on the cloud.

- **Authentication and trust of acquired information:** Because the critical data is stored on the Cloud provider's infrastructure, the owner has no control over it.

The owner will be able to use and examine the newly altered data to help make critical decisions. The validity of data is essential in this case and must be verified. Furthermore, since there are no standardized procedures in place, Data integrity cannot be guaranteed.

- **System monitoring and logs:** As users move more mission-critical apps to the Cloud, they may request that cloud providers offer more monitoring and log data for client employees. Not all Cloud providers are prepared to share portions of such data with customers or third-party auditors since the results of monitoring and logging may include sensitive infrastructure information and are typically utilized internally by providers. Cloud providers and users will need to negotiate extensively to include suitable monitoring and log information as part of any service agreement.

2.6.3 Emerging Security Challenges in Fog Computing

Fog computing extends Cloud services to the network edge. It speeds up event reaction time. It supports mobile devices that communicate via various methods. Sub-networks may use separate protocols. Thus, Fog computing has several obstacles. Here are some of the open difficulties for Fog computing:

- **Scalability:** With the dynamic development in applications and services, it is possible that a scalability problem can arise, which must be identified and remedied as soon as possible.

- **Privacy and security:** Fog nodes are located near end users. They can gather and retain private and sensitive data from users such as location, use, etc. Accordingly, Data protection is required if a fog node fails. Additionally, it is challenging to authorize and authenticate a large number of fog nodes. Approaches for dynamically evaluating the security of IoT applications are needed.

- **Dynamicity:** Without requiring any information, any edge device, sensor, or actuator may leave or join the network dynamically. As a result, network configurations are sometimes dynamic. An Automatic reconfiguration is a must-have.

- **Fault detection and tolerance:** As the volume of IoT equipment grows every day, many kinds of bugs can occur. This bug can do not appear in small-scale testing settings. Due to the scalability and dynamic elements, several fault combinations are possible. To address this problem, redundant apps are necessary.

- **Complexity:** An increasing number of manufacturers are entering in manufacturing of heterogeneous sensors and smart devices. These devices may operate on a variety of software platforms. It is becoming more challenging to find an appropriate component for all devices since certain programs can only operate with a specific hardware device and not with others.

There are presently numerous outstanding Cloud and Fog computing security issues that Cloud providers and Fog Computing should answer to persuade end-users to adopt these technologies. The most significant issues, in our opinion, are ensuring user data confidentiality while it is stored in Cloud systems or in Fog nodes that are shared among users. Indeed, the data owner or end-user cannot identify security measures to protect data in the Cloud or Fog node when the provider or the node is opaque.
Cloud computing or Fog computing offloads most IT infrastructure and data storage to third-party providers located off-premises, resulting in two significant implications [22]: (a) Because data owners have limited control over the IT infrastructure, they must establish a mechanism to enforce their security policies and ensure the confidentiality and integrity of their data, and (b) Cloud service providers have excessive privileges, giving them extensive control and the ability to modify users' IT systems and data.
These factors contribute to a low degree of confidence when storing and sharing data in the Cloud or Fog computing, even more so when a business model demands stringent se-

cure data processing to protect company interests. As a result, a secure system is critical for enabling reliable data sharing through untrustworthy fog nodes or Cloud providers. Additionally, the system should enforce data owners' access control rules, prohibiting Fog storage or Cloud storage providers or unauthorized users from accessing the data unlawfully. The following summarizes the particular security needs for data sharing in Cloud and Fog computing.

- **Security and privacy:** This is conceivably the most challenging obstacle in the data-sharing domain. In recent years, one of the most significant research fields of data sharing has been security and privacy. Indeed, externalizing data to Cloud storage can expose users' personal information to a risk of leakage or monitoring their behavior patterns, activity monitoring, interests, and preferences.
- **Access control:** It is a critical security issue that Cloud providers and customers alike must address through their data sharing networks. The Access control topic encompasses various security concerns, including data leaks, data recovery, identification and permission management, credential assignment, and user revocation.
- **Limitation of resources:** Indeed, with current technologies, users are increasingly using heterogeneous devices such as smartphones or sensors. However, these devices are limited in terms of power and energy. Therefore, it is essential to develop lightweight protocols that adapt to this resource constraint.

2.7 Cloud-Fog Computing and Resource-Constrained Devices

Today, an increasing number of "heterogeneous devices" generate and consume vast amounts of data. Numerous of these "heterogeneous devices" are components of much more extensive systems, which need computing and storage capacity to analyze and store

their data. Moreover, most of these resource-constrained devices are extremely basic, which means they will lack the necessary computing and storage capacity. In other words, external resources are needed to carry out a significant portion of data processing operations. However, the method for deploying such "external resources" is not obvious. Demand for computing and storage resources will most likely come from tens of billions of fixed and mobile endpoints spread over large geographical regions and organized in a variety of different ways, spanning a wide range of other use cases and scenarios. As a result, many of these settings will have strict requirements, such as low latency, high throughput over short periods, quick decision-making based on real-time analytics, and various combinations of these and other needs. In short, the resource-constrained devices requires substantial computing and storage resources such as Cloud computing. Indeed, Cloud computing demonstrated its effectiveness as a powerful tool for storing, processing, and analyzing data while fulfilling high-level application requirements. Unfortunately, the resource-constrained devices requirements and design space make Cloud computing impractical in many situations, mainly when creating a global and adaptable platform that can support a wide range of resource-constrained devices applications. In other words, the Cloud's centralized structure makes it challenging to satisfy the latency requirements of specific developing resource-constrained devices applications. To address this challenge, Fog computing was developed as a way to extend Cloud computing capabilities to the network's edge. This novel design combines network edge devices to address various Cloud computing bandwidth and latency constraints. Indeed, Fog computing may significantly decrease latency while maintaining the required interoperability and dependability since they are closer to end-users than Cloud data centers [23]. Moreover, Fog computing is better equipped to handle the increase in connected devices and the need for resource-constrained devices since it optimizes computing and storage resources located at the network's edge. However, Fog computing is not meant to be a rival to the Cloud; instead, it is envisioned as the perfect companion for a wide variety of use cases and applications

when conventional Cloud Computing is inadequate. Because only an intelligent mix of communications, orchestrations, and the assignment of computing and storage resources can satisfy the needs of resource-constrained devices, these two technologies are said to interact and collaboratively benefit from one another. Consequently, depending on the technical requirements and limitations of resource-constrained devices applications, it is up to the platform designer to decide whether an endpoint should be serviced by the Cloud, the Fog, or an appropriate mix of the two.

2.8 Data Sharing in Cloud and Fog Computing

Data sharing is essential for many individuals, and it is a necessary need for companies trying to increase their productivity [24]. As a result, there is a critical need for developing data-sharing applications, particularly for mass communication. However, the primary problems with such applications are security and privacy [25]. Also, data sharing in cloud and fog environments introduces new security challenges related to many aspects like data storage, access control, data confidentiality,user revocation, etc. [26].

The following section identifies the entities participating in a cloud-based or fog-based data sharing service and describes the threat model.

2.8.1 Cloud Based Data Sharing Model

By relying on cloud computing platforms, cloud service providers allows their customers to upload, access, backup, and share data over the network [27]. Currently, cloud computing platforms often feature applications for data sharing like cloud storage, cloud social networking, and cloud health [28]. Cloud sharing services are composed from three distinct entities [29]: a cloud service provider (CSP), a data owner, and a data consumer. CSP is responsible for storing and sharing outsourced data. The data owner entrusts cloud servers with their data. The customer presents their access privileges to the CSP

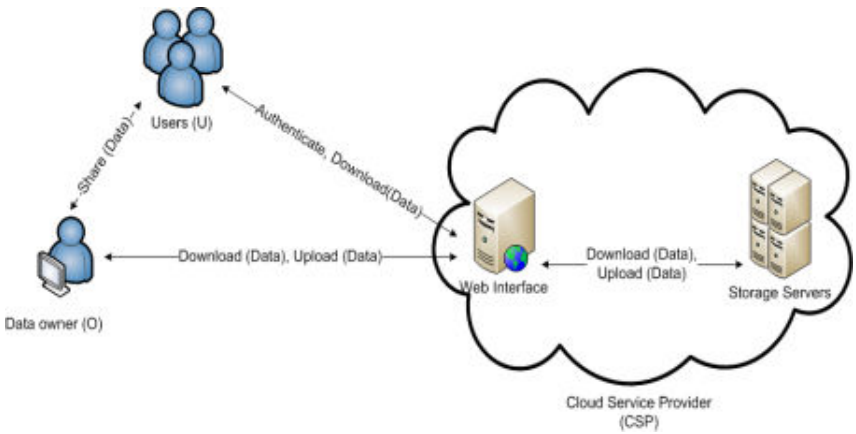

Figure 2.3: A Cloud Data Sharing Scenario[29]

to request the outsourced data.

A typical Cloud environment scenario is shown in Figure 2.3. A data owner (DO) uploads the data to the cloud computing platform and can either expose the shared data or specify the user who can access this data. Authorized users can easily access data uploaded by the DO through the cloud service provider.

2.8.2 Fog Based Data Sharing Model

There are four entities in fog-based data sharing[30]: Data Owner, Cloud Servers, Fog Nodes, and Data users .

- The Data Owner (DO) has the authority to access and change the data. He encrypts the data and transfers the encrypted data to the cloud servers.

- The Cloud Server (CSP) is in charge of data storage and its distribution to the fog nodes.

- Fog Nodes (FNs) serve as intermediate entities between the cloud and users. they can perform intermediate operations on the data and make these data closer to the user thus optimizing both network storage and computing resources.

- The term "data users" refers to people or devices who request data access if they have

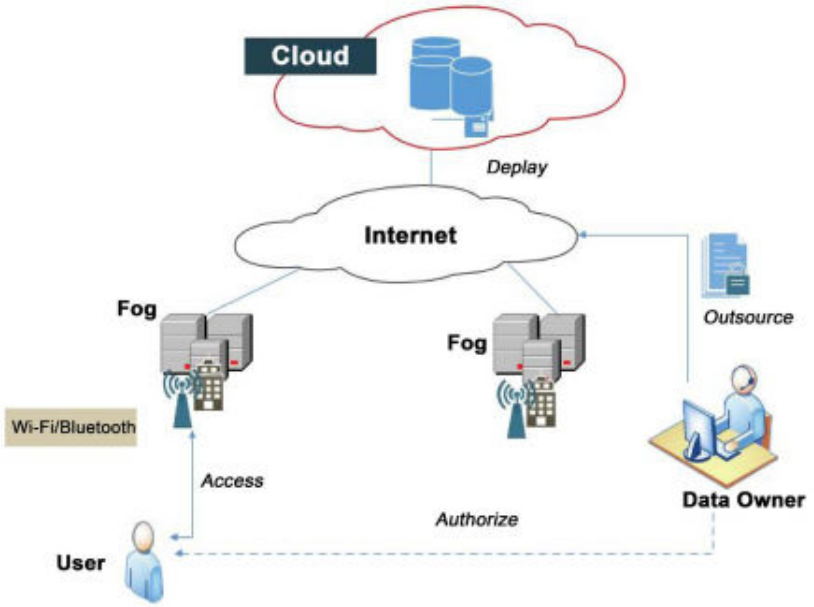

Figure 2.4: A Fog Data Sharing Scenario[30]

the necessary permissions.

Figure 2.4 depicts a fog environment scenario in which a DO encrypts a data file and subsequently outsources its storage to a CSP. The CSP then distributes the data file to the appropriate fog node. Fog nodes are geographically spread and have fixed positions within a specific domain. The user can request the data from the fog node closest to him. The fog node accepts the user's request and forwards it to the appropriate file.

Security challenges to cloud-fog data services are presented by two categories of adversaries, external (e.g., hackers) and internal (e.g., curious CSP or fog node). An external adversary is a malicious user who employs attack tactics such as network eavesdropping, vulnerability scanning, or malware to get unauthorized access to cloud data. In general CSP or fog can be considered as internal adversaries and cannot be trusted entirely. The CSP or fog node are trustworthy yet curious in most present cloud or fog security solutions.

2.8.3 Discussion

In summary, Cloud computing can increase privacy concerns, mainly due to the lack of openness about data processing and storage, the dynamic nature of the Cloud, and the absence of mechanisms for enforcing privacy policies in such an environment. As a result, it is understandable that Cloud clients' worries regarding their sensitive outsourced data are growing. Privacy concerns are a significant barrier to Cloud adoption and may incur fear of data outsourcing. Thus, it is suggested to pay more attention to this element. Several mechanisms have been proposed to address Cloud privacy concerns. The following section examines many existing mechanisms.

2.9 Security Mechanisms

The term "Security mechanisms" refers to the technological methods and techniques used to implement security services. A mechanism can operate independently or in conjunction with others to provide a specific service. This section discusses three security mechanisms in detail: Cryptography, Access control, and Cryptography Access control.

2.9.1 Cryptography

In recent years, the art of hiding messages, or cryptography, has developed into a precise science. It integrates various disciplines, including mathematics, computer science, and even physics, to safeguard data by offering multiple security services such as authentication, integrity, confidentiality, and non-repudiation. Cryptography is classified into two types: Symmetric cryptography and Asymmetric cryptography.

Symmetric Cryptography: The oldest and most well-known cryptographic technique is symmetric encryption. It entails combining a shared secret key with a message to modify its content in a specific way. This shared key can be used to securely exchange

messages as long as both the sender and the recipient know the secret key.

Symmetric key ciphers are useful for the following reasons:

- Producing a strong key for the ciphers does not require a high computational cost.
- The keys are usually much smaller in size than the security they provide.
- These methods are pretty fast when they come to encrypting and decrypting with the encryption/decryption algorithms.

The primary disadvantage of symmetric encryption is the difficulty in sharing the secret key since any exchange must preserve its privacy. This usually requires transmitting the secret key over a protected channel or encrypting it with a different key, resulting in a never-ending reliance on another key. Additionally, a user must create a new secret key for each contact with another user. As a result, he will store as many keys as possible to users with whom he has developed contacts.

Asymmetric Cryptography: A cryptographic scheme in which each person is associated with a pair of keys is known as public-key cryptography (public key and private key). In asymmetric cryptography, when one part of the key is used to perform a cryptographic operation, the other part is used to complete the opposite process. For example, the public key is used for encryption, while the private key is used for decryption, which is the case for signatures where the private key is used for signature, and the public key for verification [31]. Asymmetric algorithms are preferable to symmetric algorithms for the following reasons:

- They avoid the key distribution problem and speed up key management by using public and private keys.
- They increase protection because private keys are never exchanged or exposed to anyone.

- They establish the non-repudiation of the transaction.

The primary disadvantage of public-key cryptography is that it requires more computing time than symmetric cryptography. As a result, using the form of encryption with large volumes of data is not always sufficient. However, an intriguing solution would be to submit a symmetric key via public-key encryption, used in subsequent data encryption operations.

2.9.2 Access Control

Access control is a form of a security strategy that aims to limit who or what has access to or uses resources in a computing environment. Physical and logical access control are the two forms of access control. Physical access control systems govern physical entities and facilities such as campuses, offices, rooms, and physical IT assets. In contrast, logical access control controls computer networks, system resources, and data. An access control system's primary components are an access right and two entities known as the subject and object.

- A subject may be a user, method, thread, or program wishing to perform specific actions within a system.
- An object is a physical entity within a system upon which a user may perform actions.
- Access rights define the conduct that a subject is permitted to perform on a particular object.

Due to the primary objective of this study being to ensure data protection in a Cloud environment, we will concentrate exclusively on the logical access control mechanisms presented in the following sections. To safeguard the objects contained within a logical access control scheme, security administrators implement an access control mechanism

that can be described as "The logical component that serves to receive the access request from the subject, to decide, and to enforce the access decision." [32]. These systems are usually based on the following access control models:

1. **Discretionary Access Control (DAC):** DAC is an access management model in which access rights are determined at the discretion of the object's owner or any entity that controls access to the object.

2. **Mandatory Access Control (MAC):** MAC is a model of access control in which a central authority regulates access rights based on various levels of protection.

3. **Identity-Based Access Control (IBAC):** IBAC is a control access model in which the system stores the identities of those granted access to an object using mechanisms such as access control lists (ACLs).

4. **Role-Based Access Control (RBAC):** RBAC is a form of access control model in which the device assigns each subject a predefined role with a distinct set of privileges.

5. **Attribute-Based Access Control (ABAC):** ABAC is a form of access control model in which each user is assigned specific attributes. ABAC systems determine access rights based on the characteristics of users, systems, and environmental factors.

Many cloud access control schemes have different descriptions of rules, which cannot satisfy the application requirements due to many shortcomings. In addition, the use of access control alone does not protect the confidentiality of data against cloud computing providers. Consequently, using a cryptography mechanism is essential to protect data since it offers multiple security services, such as the confidentiality of data. However, cryptography alone does not provide access control, i.e., the users who can access the data.

For this, A new paradigm has been introduced to ensure data security it is Cryptographic Access Control.

2.9.3 Cryptography Access Control

Cryptographic access control is a paradigm for federating information systems globally. This model is an access management scheme entirely reliant on cryptography to ensure the security and integrity of the data handled by the framework. Additionally, it enables secure access control in untrustworthy environments where a lack of global information and control are defining characteristics. The following sections discuss some of the most sophisticated cryptographic access control techniques.

2.9.3.1 Identity Based Encryption (IBE)

IBE is a sophisticated public-key encryption technique [33] that generates a user's public key from identity information such as the user's email address. In IBE, a trusted central authority generates system parameters such as a public/master pair of keys, message/ciphertext spaces, etc. It publishes some of the generated system parameters (public parameters). A sender with access to the system's public parameters can encrypt a message using the receiver's unique identifier (email address, for example) as a key. On the other hand, the receiver must get its decryption key from the central authority that sets the public parameters to decrypt the obtained data successfully.

2.9.3.2 Fuzzy Identity Based Encryption (FIBE)

FIBE is a modern form of Identity-Based Encryption (IBE) scheme that utilizes public-key encryption. The user's identity is viewed as a set of descriptive attributes in FIBE [1]. The FIBE scheme permits a private key for an identity θ to decrypt a ciphertext encrypted with an identity θ', if, and only if the identities θ and θ' are adjacent. Closeness is determined in this cryptographic scheme using the set overlaps distance metric. For

	Method	Advantages	Disadvantage
Identity Based Encryption (IBE)	An ID-based primitive, this is a type of public key encryption in which a user's public key is unique information about the user's identity.	Eliminates the need for a public key distribution infrastructure.	Requires a trusted authority that indicates that the message was confidential
Fuzzy Identity Based Encryption (FIBE)	Uses IBE with biometric identities as attributes such as iris scan attributes.	The main advantage of fuzzy encryption is error tolerance.	The Initial construction is limited in terms of the expressibility of who can decrypt the cipher-text, as the attribute formula consisted of a threshold gate.
Attribute-Based Encryption (ABE)	Encryption is based on a set of attributes, describing data properties, user properties,and environment properties, as well as an access structure indicating who can access what.	- Fine grained access control. - It does not depend on key sharing or key management algorithms. - Anti-collusion. - One-to-many user encryption. - Allowing access only to recipients who meet predefined attributes.	The computational cost during the encryption and decryption phases increases exponentially with the complexity of the access policy.

example, to decrypt a message, C is encrypted with the public key θ', we need a private key for the identity θ with $|\theta \cap \theta'| \geq d$.

2.9.3.3 Attribute Based Encryption (ABE)

The ABE technique expands identity-based encryption by offering varying permissions for people to access expression-encrypted files. ABE enables one-to-many encryption. It is imagined as a promising cryptographic primitive for enforcing flexible and fine-grained access control schemes that allow users to share their data according to their encryption policy without understanding who would receive it. Access is allowed depending on a list of attributes. The two primary variants of ABE are Key-Policy Attribute-Based Encryption (KP-ABE) [34] and CiphertextPolicy Attribute-Based Encryption (CP-ABE)[35].

2.9.3.4 Discussion

IBE (identity-based encryption) is a public key encryption in which a user's public key contains unique information about the user's identity. This method eliminates the need for a public key distribution infrastructure. However, this method requires a trusted authority that indicates that the message was confidential. In 2005, Sahai and Waters [1] proposed fuzzy IBE, which could be used for biometrics and has the property of error tolerance. However, the Initial construction in this method is limited in terms of the expressibility of who can decrypt the cipher-text, as the attribute formula consisted of a threshold gate.

Attribute-based encryption is an attractive research topic because it provides fine-grained, anti-collusion, one-to-many encryption and does not depend on key sharing or key management algorithms. Nevertheless, this technique has some drawbacks as computational cost during the encryption and decryption phases that we detail in the next chapter.

2.10 Conclusion

In this chapter we have discussed the fundamental aspects of Cloud and Fog computing. We provides brief description on some security concepts related to our work. Using cloud and fog computing for data sharing is inevitable, however it introduces several security issues including data privacy. Indeed by outsourcing data to the Cloud server, the data owner loses physical control over his their sensitive data. Data sitting on a Cloud server is more vulnerable to hostile insider and outsider attacks. Therefore, we concentrated on cryptography and access control for the Cloud's safe and secure data storage. We also showed that one of the prominent techniques to ensure fine-grain access control in resource-constrained devices is Attribute-Based Encryption. However, some challenges are facing the deployment of this technique. The following chapter highlights those challenges

and survey proposed solutions that pave the way to adopt Attribute-Based Encryption as a primary technique for ensuring access control in data sharing.

Chapter 3

Data Sharing for Resource-Constrained Devices Based on ABE

3.1 Introduction

Attributes-based encryption is an encrypted access control mechanism that ensures efficient data sharing among dynamic groups of users by setting up access structures indicating who can access what. This mechanism allows users to encrypt and decrypt messages against a set of attributes. It is a powerful and promising cryptographic solution that permits to keep the encrypted data confidential even when the storage server is untrusted. Nevertheless, in ABE, encryption and decryption are time-consuming which is a considerable limitation when devices have limited CPU power, memory, and energy [36]. Another drawback is that the access policy is sent in the clear text along with the cipher-text, which could allow a malicious user to compromise privacy by exploiting the sensitive information (like social security number, name, etc.) contained in the access policy. Many existing ABE-based approaches try to address issues like computation overhead and policy hiding. However, these approaches are very limited in terms of user collaboration as they only allow assigning one access authorization to one user. However, in some cases,

the encrypted data cannot be granted individually for one user but to a group of users with the right access attributes which are used in a collaborative process to access the data. This chapter presents Attribute-Based Encryption (ABE), which offers collusion-resistant and fine-grained data access control. Then, we present the Two main variants of attributes-based encryption. Afterwards, we present the challenges of implementing ABE and its implementation in the context of resource-constrained devices. We also present existing solutions that try to address these challenges. We summarize the related work on ABE, considering the difficulties introduced by the performance of ABE and their implementation in the context of the resource-constrained device. We analyze and compare the existing work against a set of criteria and point out their limitations.

3.2 Attributes Based Encryption

3.2.1 Overview

Attribute-Based Encryption (ABE) is a public key encryption mechanism that was proposed by Shai and Water in [1]. It allows one to many encryptions and keeps the encrypted data confidential even when the storage server is untrusted. In ABE, each user is attached to a descriptive string named attribute; several attributes can also characterize the user. These attributes describe data properties, user properties, and properties of the environment. Consequently, this technique allows the data owner to implement access control by setting up access structures indicating who can access what. In ABE, the user can decrypt the cipher-text if and only if the attributes of the cipher-text satisfies the attributes in the user's key. The user who wants to share the data (the Data Owner (DO)) executes the encrypted algorithm. In the beginning, the DO defines an access policy in the form of an access tree (Figure 3.1) then encrypts the data to have a cipher-text. This last is shared and stored in Cloud storage servers where it is accessible for everyone. However, only users authorized by the access policy defined by the DO can decrypt the encrypted

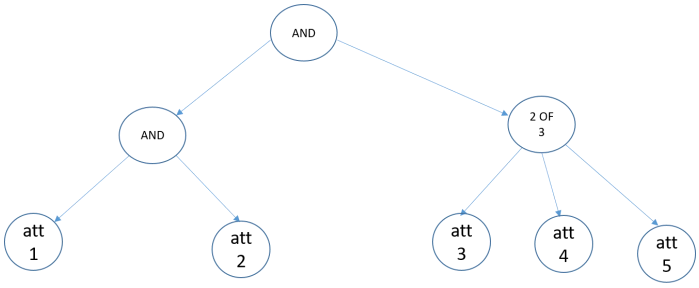

Figure 3.1: An Example of an Access Policy

text by executing the decrypted algorithm.

3.2.2 Preliminaries

3.2.2.1 Bilinear Maps

Let G_1,G_2 and G_T be three cyclic groups of prime order p and let g_1 be a generator of G_1 and g_2 be a generator of G_2. A bilinear map is a map $e : G_1 \times G_2 \rightarrow G_T$ with the following properties:

- Bilinearity: $\forall g_1 \in G_1, g_2 \in G_2$ and $x, y \in Z_N$, it satisfies $e(g_1^x, g_2^y) = e(g_1, g_2)^{xy}$.
- Non-degenerate: $\exists g_1 \in G_1$, $g_2 \in G_2$ such that $e(g_1, g_2) = 1$.

  Also notice that the map e is symmetric because $e(g_1^x, g_2^y) = e(g_1, g_2)^{xy} = e(g_1^y, g_2^x)$

3.2.2.2 Access Tree

Let T be a tree representing an access structure. Each non-leaf node of the tree represents a threshold operator described by its children and a threshold value. If num_x is the number of children of node x, and k_x is its threshold value, then $1 \leqslant k_x \leqslant num_x$. When $k_x = 1$, the threshold is an OR operator, and when $k_x = num_x$, it is an AND operator. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$ [35].

Let T be an access tree with root r. T_x denotes the subtree of T rooted at node x. Thus, T is the same as T_r. If a set of attributes ω satisfies the access tree T_x, we denote

it as $T_x(\omega) = 1$. We compute $T_x(\omega)$ recursively as follows. If x is a non-leaf node, we evaluate $T_x(\omega)$ for each child x of node x. $T_x(\omega)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $T_x(\omega)$ returns 1 if and only if $att(x) \in \omega$, where $att(x)$ denotes the attribute associated with node x [35].

3.2.3 ABE Algorithms

Two main variants of attributes-based encryption exist, namely, Key-Policy Attribute-Based Encryption (KP-ABE)[34], and Ciphertext-Policy Attributes Based Encryption (CP-ABE)[35]. Both types of ABE scheme consists of four fundamental algorithms: setup, encrypt, key generation, and decrypt. The setup algorithm and the key generation are executed by a particular entity called a Trusted Authority (TA). The TA performs, in the beginning, the setup algorithm to generate a Public Key (PK) and Master key (MK) and that according to the algorithm type using CP-ABE (section 3.2.3.1) or KP-ABE (section 3.2.3.2). The TA also performs the keygen for each user to generate the user's private key called Secret Key (SK), where this key is based on the set of user attributes. The encrypt algorithm and the decrypt algorithm are executed respectively by the DO and the who wants to access data.

3.2.3.1 CP-ABE

In CP-ABE, the attributes describing the user's characteristics' information are associated with the user's private key. The data is encrypted with the access policy formulated by the data owner. A user can decrypt the data if an only if its attributes satisfy the access policy defined by the data owner.

In the example in Figure 3.2, the data owner creates an access structure defining who can access the encrypted data, which is stored in the Cloud. In the example, only User3 can decrypt the data because he has the attribute {President}. User1 and User2 cannot decrypt the message because their attributes Bob, Staff, Seoul and Alice, Manager,

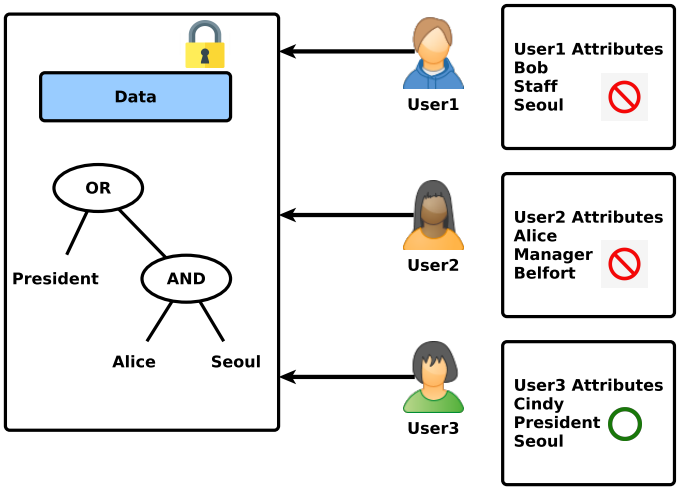

Figure 3.2: CP-ABE-Ciphertext-Policy-Attribute-Based-Encryption[37]

Belfort cannot satisfy the "AND" of the access policy.

The scheme of CP-ABE consists of the following four algorithms [35].

Setup (U): The setup algorithm is performed by the trusted authority TA which takes an attribute universe U as input. It chooses a bilinear group G_0 with order prime r and g as a generator of G_0, for each attribute labeled from 1 to $|U|$ the authority choose $h1...h|U|$ uniformly at random from G_0 where the number is used to index the attributes. Then it will choose two random exponents $\alpha, a \in Z_p$. It outputs the public key PK published as:

$$PK = \{(g,g)^{\alpha}, g, g^{a}, \{h_i\} \forall i \in U\}$$

And the Master Secret Key (MK) by:

MK= {a,α}

All the entities know the public key of the system. Whereas the Master Secret Key is kept secret.

Keygen(PK, MK, ω) : The TA runs this algorithm. It takes as input the public key PK, the master key MK, and a user's attribute set ω. The TA chooses a random $t \in Zr$. It will output a private key SK as follow:

$$SK = \{k = g^{(\alpha+at)}, L = g^t, \{K_j = h_j^t\}\forall j \in \omega\}$$

Encrypt(PK, M, T) : This algorithm is executed by the entities who want to share and encrypt the message. It takes as input the public key PK, a message M, and an access policy tree T. It will produce a cipher-text CT, which is the encrypted message with the access policy T embedded.

First, the algorithm chooses a polynomial $q(x)$ for each node x in the policy tree T. These polynomials are chosen from top to down, starting from the root node r. For each non-leaf node x in the tree, the degree $d(x)$ is defined by $d(x) = k(x)1$. Then, starting with the root node of the policy tree, the algorithm randomly chooses $s \in Z(p)$ and sets $q_{r(0)} = s$. After that, it randomly chooses $d(r)$ other points of the polynomial q_r to define it entirely. Subsequently, it sets for other node x (including both leaf nodes and non-leaf nodes) $qx(0) = q_{parent(x)}(index(x))$, and randomly chooses dx other points and completely defines q_x. The degree of leaf nodes is set to be 0. Once all the polynomials have been defined, we put $\lambda_x = q_x(0)$ for each x in T. In addition, we choose a set of n random number $r_1...r_n$ where each $r_i \in Z_p$. For each λ_i, r_i the data owner computes $C_{(1,i)} = g^{\lambda_i a}.h_{\pi(i)}^{-r_i}, C_{(2,i)} = g^{r_i}$. We note that g^a and $h_{\pi(i)}$ are present in the public key. Finally, the cipher-text is defined by:

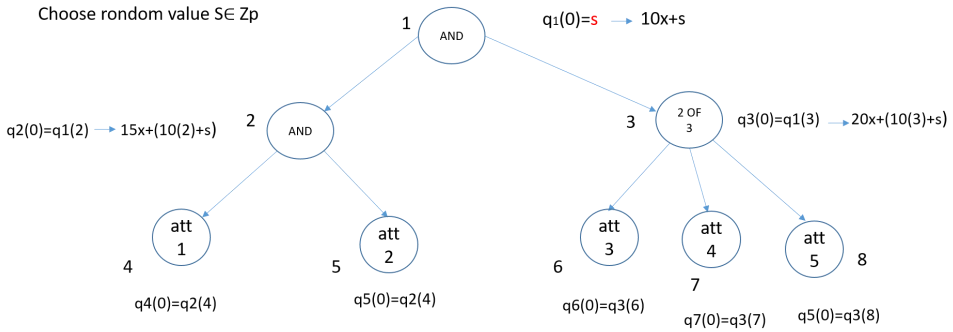

Figure 3.3: An Example of how the secret S is shared in an Access Policy

$$CT = \{T, C_0 = Ke(g,g)^{s\alpha}, c^{'} = g^s, \{C_{(1,i)}, C_{(2,i)} \forall i\}\},$$

The user can decrypt the CT only if his secret key satisfies the access policy.

Decrypt (CT, SK): The decryption algorithm is executed by who wants to access data. It takes a private key SK and a cipher-text CT as input. It will output the plaintext M if attributes set ω upon which SK is constructed satisfies T.

The user can compute a set of $c_i^{'}s \in Zr$ by applying Gauss-Jordan elimination such that $\sum c_i \lambda_i = s$. Recall that $\lambda_i^{'}s$ are shares of secret s. The user compute:

$$e(C^{'}, K) / \prod_a \left(e(C_{1,i}, L) e(C_{2,i}, K_{\pi(i)})\right)^{C_i}$$

$$e(g,g)^{s\alpha} e(g,g)^{ast} / \prod_a (e(g,g)^{ta\lambda_i})^{C_i} = e(g,g)^{s\alpha}$$

Then the user execute $M = C_0/(g,g)^{s\alpha}$ to recover original message M.

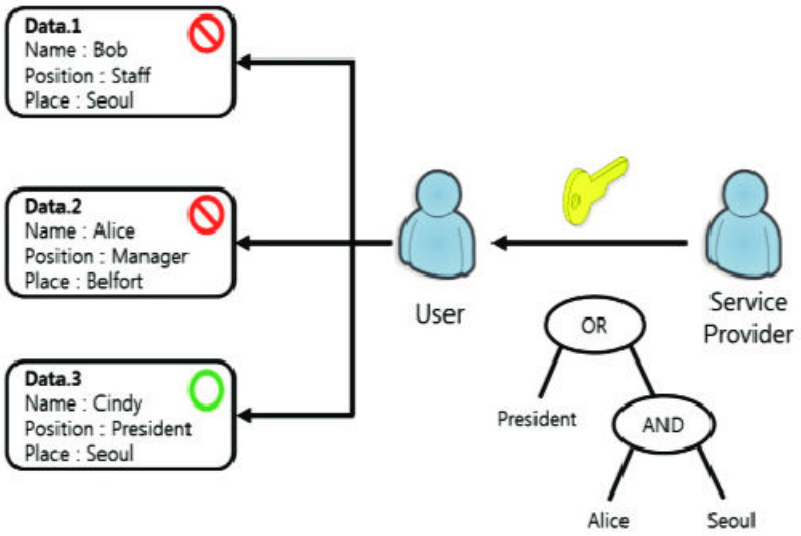

Figure 3.4: KP-ABE-KEY-Policy-Attribute-Based-Encryption[37]

3.2.3.2 KP-ABE

In KP-ABE (Figure 3.4), in contrast to CP-ABE, the set of attributes is associated with the cipher-text, and each private key is associated with the access policy formulated by the data owner. For example, shared data encrypted with the attributes "Alice" and "Seoul" Or "President" can be decrypted by the user who has in his private key the access formula "President","Alice", and "Seoul". Therefore the user with access formula with Alice, Manager, and Belfort will not be able to decrypt the encrypted data.

Setup: The setup algorithm is performed by the trusted authority to create the Public and the Master Keys. It starts by choosing a bilinear group G_0 with order prime r and g as a generator of G_0. Next, the TA defines the universes attributes for each attribute labeled from 1 to $|U|$ the authority chooses $h1...h|U|$ uniformly at random from G_0 where the number is used to index the attributes. The primitive also chooses a random number y from Z_p.

Finally, the public parameters are published as:

$PK = (T_1 = g^{t_1}, , T_{|U|} = g^{t_{|U|}}, Y = e(g,g)^y)$
And kept $(t_1, , t_{|U|}, y)$ secret as a Master Key MK.

Keygen (T, MK): The TA runs this algorithm. It takes as input an access tree T and the master key MK. First, the algorithm chooses a polynomial $q(x)$ for each node x in the policy tree T. These polynomials are chosen from top to down, starting from the root node r. For each non-leaf node x in the tree, the degree $d(x)$ is defined by $d(x) = k(x)1$. Then, starting with the root node of the policy tree, it sets $q_{r(0)} = S$. After that, it randomly chooses $d(r)$ other points of the polynomial q_r to define it entirely. Subsequently, it sets for other node x (including both leaf nodes and non-leaf nodes) $q_{x(0)} = qparent(x)(index(x))$ and randomly chooses d_x other points and completely defines qx.

Finally, for each leaf node x, the authority gives the following private key to

$D_x = g^{\frac{q_{x(0)}}{t_i}}$

where $i = att(x)$.

Encrypt (M, y, PK): This algorithm is executed by the entities (data owners) who want to share and encrypt the message. The data owner (DO) chooses a random $s \in Z_p$ and encrypts the message$M \in G_T$ under attributes set y, and then it computes the cipher-text as:

$CT = (y, MY^s, \{T_i^s\}_{i \in y})$

Decrypt (CT, D): The decryption algorithm is executed by who wants to access data. It takes a private key D, which contains the access policy and a cipher-text CT as input. If this entity wants to decrypt the cipher-text that contains attributes set w that supposed satisfies the access policy, then for each leaf node x, the entity computes:

$$e(D_x, T_i^s) = e(g^{\frac{q_{x(0)}}{t_i}}, g^{st_i}) = e(g,g)^{sq_{x(0)}} \text{ for attr(x) =i}$$

Knowing that $q_{x(0)} = p_{parent(x)}(index(x))$ and using polynomial interpolation the user can compute $e(g,g)^{sq_{parent(x)}}$for node x. The user repeats this approach until reaches root node r. The user obtains $e(g,g)^{sq_{r(0)}} = e(g,g)^{sy}$ (if and only if the attributes in the cipher-text satisfies the tree in the private key) and he can recover the original message M by dividing MY^s by $e(g,g)^{sy}$.

3.2.4 Discussion

Many researchers have focused their efforts on ABE due to its high scalability and its access control mechanism over outsourced data. On the other hand, CP-ABE has received far more attention than KP-ABE. The reason for this is that the data owner is in charge of determining the access policy in CP-ABE. Indeed, instead of creating an access policy for each user with their private key, it will only create an access policy for the encrypted document. This means that if we want to revoke users, it suffices to modify the access policy and only re-encrypt the document without recreating all the user and the access policies. On the other hand, KP-ABE is better indicated for media diffusion or streaming. For example, suppose we want to encrypt a cartoon musical movie, which can be watched by child users when we use KP-ABE. In that case, we can encrypt movies with cartoon and musical attributes, then generate the secret key with cartoon AND musical structure attribute for kids. Nevertheless, if we use CP-ABE, we have to think about how only

users assigned with cartoon AND musical can decrypt and then generate the secret key with cartoon and musical attributes for child users.

In our work, we chose to use the CP-ABE because it has the advantage of being flexible in the management of attributes and scalable in terms of processing multiple levels of attribute authorities, unlike KP-ABE.

3.3 Challenges of Implementing ABE in Resource-Constrained Devices

In this section, we survey some challenges confronting ABE implementation in resources-constrained devices. We consider mainly three challenges: How to overcome the decryption overhead on a resource-constrained device? How to protect the confidentiality of the access policy so that the malicious user cannot exploit sensitive information in the access policy? How to achieve a controlled collaboration in the same user group to satisfy the access policy and to access the data?

3.3.1 Resource Limitations

ABE's pairing and exponentiation operations generate complexity and a heavy overhead in the resources-constrained devices. These latter have very limited resources in terms of energy, storage, and computing capacity. These inconvenience makes ABE mechanisms impractical, which is a significant issue for keeping privacy in such resource-constrained contexts. Indeed, this infeasibility motivates the researchers to propose many techniques and solutions to reduce ABE schemes overhead, such as computation outsourcing, compression, etc.

3.3.2 Hidden Policy

In CP-ABE, the data is encrypted with the access policy. This last is sent in the clear text along with the cipher-text, which could allow a malicious user to compromise the legitimate user's privacy by exploiting the sensitive information (like social security number, name, Etc.) contained in the access policy [38].

3.3.3 Collaboration

ABE-based approaches try to address issues like computation overhead and policy hiding. However, these approaches are very limited in terms of user collaboration as they only allow assigning one access authorization for one user that can satisfy the access policy. However, in some cases, the encrypted data cannot be granted individually for one user. Still, a group of users with the right access attributes can collaborate to access the data.

3.4 Existing Data Sharing Solutions Based ABE

In this section, we present existing solutions that try to address the challenges mentioned in the previous section.

3.4.1 Outsourced Computation

To reduce the computation overhead in constrained devices, several CP-ABE schemes with outsourced computation have been proposed. Green et al.[39] propose a scheme where a transformation user key is derived from the user's secret key, which allows outsourcing the heavy decryption operation to the Cloud. Zhou et al. [40] proposed a new CP-ABE scheme, in which the encryption and decryption process is outsourced on external Cloud based services. In the encryption process, the authors connect two access structures T1

and T2 to form a single access policy. A root AND node connect these access policies. The first part of the encrypted text is generated by sending T1 to an external encryption service while the second part is computed by the user using T2, where this T2 contains only one attribute. However, one flaw in this approach is that the access policy in this scheme is not hidden.

In their work Touati et al.[41] present a cooperative CP-ABE for the Internet of Things, where the complex operations of the CP-ABE encryption primitive forced authors to use intermediates Unconstrained devices to outsourced encryption process. The authors assume that unconstrained devices are trusted. In this scheme, the data owner (device A that is a resource constrained device) encrypts the data under access T. During the process; device A is supported by a set of secure assistant devices that perform the exponentiation operation instead of the device A itself. The authors suppose that the intermediate unconstrained devices are trusted, but they do not suggest externalization of the decryption process, another drawback is that the access policy is sent in the clear on the network, where the access structure can also contain some sensitive private information.

In their paper, Yang et al.[42] use multi-authority CP-ABE schemes that support outsourced decryption. In [43], Li et al. present a scheme that requires two Cloud service providers to perform outsourced decryption algorithms. The research work in [44] introduces an approach that permits to construct ABE schemes with verifiable outsourced decryption.

In [45], the authors propose a new method for outsourcing CP-ABE, namely the EOEB (outsourcing mechanism for the encryption of the ABE encryption policy). The main idea is to reduce encryption costs by delegating the most intensive computations of the encryption phase of the CP-ABE to a semi-trusted party. The authors divide the encryption process in to two phases: the Pre-delegation phase, and the compDelegation phase. Pre-delegation is performed by KDG (Key delegation) which executes the con-

figuration algorithm as in the basic CP-ABE. It also generates a secret delegation key for each data producer (DP) and a list of security parameters. This list is then sent to DG (delegate). Two steps are executed in the compDelagation phase. The first step is executed by DP. In this step, the DP generates the temporal encrypted text CT' which contains the Blinded value s. The second step is executed by DG (delegate) which executes the most expensive computation operation without any knowledge of the secret message M. Nevertheless as in the work of [41], the authors do not propose to outsource the decryption process which consumes IOT energy at the user level and they do not hide the access policy.

Fan et al.[46] proposed an outsourced, secure, and verifiable multi-authority access control system called VO-MAACS, where most encryption and decryption computations are outsourced to Fog devices. In a similar work [47], the authors propose a fast decryption process using the Fogs node in a parallel scheme, in which each Fog sends -in parallel- the intermediate result to the receiver. Zuo et al.[48] present a CCA-secure ABE scheme that supports outsourced decryption for Fog computing. In [49], Yeh et al. proposed a decryption outsourced framework for health information access control in the Cloud by utilizing a Cloud Service Provider (CSP) to check whether the attributes satisfy the access policy in the cipher-text. Li et al.[50] proposed a secure method to generate the transformation key with supporting verifiable outsourced encryption and decryption process. In [51], the authors proposed an approach for privacy computing where the decryption process is outsourced to the Cloud. In addition, their work to attribute revocation. ABEM-POD [52] is a model with parallel outsourced decryption for edge intelligent Internet of Vehicles, based on Spark and MapReduce. Sabitha et al [53] proposed a multi-level on-demand access control for flexible data sharing in the Cloud, where the decryption rights of the selected set of cipher-text classes are delegated to selected users. To reduce the computation overhead on the user, part of the decryption is performed by the CSP without disclosing the attribute key details. The end user performs the remaining

decryption. The paper in [54] introduced a secure and efficient data sharing scheme where computation is outsourced to Fog nodes. The approach also supports attribute revocation through dynamic policy updating. Sethi et al [55] proposed a practical decentralized multi-authority, traceable and revocable attribute-based cryptosystem with outsourcing decryption (PMTER-ABE). PMTER-ABE aims to implement white-box traceability to detect malicious users and outsource computationally intensive decryption operations to a proxy server.

3.4.2 Hidden Policy Approaches

In their papers [56], Nishide et al. proposed an attribute-based encryption scheme with a partially hidden access control policy. In [57; 58], the authors proposed an attribute-based encryption scheme with hidden access policy. This construction is based on a simple and less expressive AND gates access structure. Sun et al.[59] proposed a CP-ABE with a simple hidden access policy. In their scheme, each attribute in the access policy can have multiple values. Zang et al.[60] presented a scheme with a partially hidden policy where the access policy related to the cipher-text includes just the attributes names. In [61], the authors proposed securely outsourcing multi-authority ABE with policy hidden for Cloud-assisted IoT (PHOABE). In PHOABE, the attributes in the access policy are hidden, and the decryption process is outsourced to the Cloud. In their papers [62][47], the authors proposed a CP-ABE scheme with a hidden access policy where false attributes are added thus preserving its confidentiality. Other works like [63], used ABE as a second layer to secure other types of access policies (written in XACML). Although CP-ABE can encrypt such policies' attributes, the access policies included inside CP-ABE's cipher-text are sent in clear text. Also, the computation is still expensive and thus is not suitable for IoT. In [64], the authors proposed VHPDT, a verifiable hidden policy CP-ABE with a decryption testing scheme. In addition, they showed its applications in VANETs.

3.4.3 Collaborative Approaches

Li et al.[65] proposed GO-ABE, a group-oriented ABE scheme. Their solution addresses the collaboration problems among users by dividing users into groups. To access the data, users in the same group can collaborate to satisfy the access policy. In GO-ABE, The collaboration between users in the same group is not limited, i.e: each user can share all of his attributes even if the data owner does not allow this kind of access. Y.xue et al.[66] proposed an attribute-based controlled access. In this scheme, the data owner controls collaboration among users in the same group by specifying a collaboration attribute that is defined in the access policy. To this end, a translation value is added to the cipher-text and a translation key is integrated inside the secret key. The combination of these two allows users to collaborate im order to satisfy the access policy.

Chen et al [67] proposed an Efficient CP-ABE Scheme with Shared Decryption in Cloud storage, where the users can collaborate to decrypt the cipher-text through the use of an integrated access tree.

3.4.4 Comparison

Table 3.1 summarizes the state-of-the-art methods with respect to (1) is the approach suitable for resources-constrained devices? i.e: is it energy consumption intensive (2) does it use Access Policy Hiding to ensure the attributes' privacy? (3) is the decryption process outsourced? And (4) Does the approach support Collaboration?. The Table shows that in most existing schemes that use outsourced decryption [42; 43; 44; 46; 47; 48; 49; 50; 51; 52; 53; 54; 55; 68], the access policy privacy is not taken into account. In addition, the collaboration between users in the same group is not considered. Also, Outsourcing approaches can be further divided into groups; Cloud and Fog outsourcing approaches. Fog approaches are more efficient since Fog devices are closer to the user and the deployment of such devices is cheaper, more resource and network efficient.

On the other hand, most research works that use a hidden access policy [56; 57; 58; 59; 60; 61; 62; 64] do not consider outsourcing decryption and collaboration in the same group and are thus not suitable for limited resource devices like in IoT.

Table 3.1: Comparison with Related Work.

Reference	Suitable for IOT	Hidden Policy	Decryption Outsourcing	Cloud/Fog Outsourcing	Collaboration	Technique
Nishide et al [56]	✗	✓	✗	✗	✗	Partially hidden access control policy
Green et al [39]	✓	✗	✓	Cloud	✗	Transformation user key & decryption outsourcing
Lai et al [57]	✗	✓	✗	✗	✗	Hidden access policy based on simple AND gates access structure
Yang et al [42]	✗	✗	✓	Cloud	✗	Multi-authority CP-ABE & decryption outsourcing
Li et al [65]	✗	✗	✗	✗	✓	Limitless attribute sharing among users in the same group

Continued on next page

Table 3.1 – continued from previous page

Reference	Suitable for IOT	Hidden Policy	Decryption Outsourcing	Cloud/Fog Outsourcing	Collaboration	Technique
Mao et al [44]	✓	✗	✓	Cloud	✗	verifiable outsourced decryption & decryption outsourcing
Fan et al [46]	✓	✗	✓	Cloud	✗	Verifiable multi-authority control & encryption/decryption outsourcing
Wang et al [62]	✗	✓	✗	✗	✗	False attributes are added to hide the access policy
Sun et al [59]	✗	✓	✗	✗	✗	Each attribute in the access policy can have multiple values
Zuo et al [48]	✓	✗	✓	Fog	✗	decryption outsourcing
Belguith et al [61]	✓	✓	✓	Cloud	✗	multi-authority ABE & policy hidden & decryption outsourcing

Continued on next page

Table 3.1 – continued from previous page

Reference	Suitable for IOT	Hidden Policy	Decryption Outsourcing	Cloud/Fog Outsourcing	Collaboration	Technique
Zhang et al [60]	✗	✓	✗	✗	✗	Partially hidden access policy that only includes attributes names
Abd El-Aziz et al [63]	✗	✗	✗	✗	✗	XCAML for authentication
Zhao et al [64]	✓	✓	✓	Fog (RSU)	✗	Verifiable hidden policy & decryption outsourcing to RSUs
Xue et al [66]	✗	✗	✗	✗	✓	Translation value in the cipher-text and translation key inside the secret key
Tu et al [54]	✓	✗	✓	Fog	✗	Encryption/Decryption outsourcing & Dynamic policy updating
Fan et al [51]	✗	✗	✓	Cloud	✗	Decryption outsourcing (Cloud)

Continued on next page

Table 3.1 – continued from previous page

Reference	Suitable for IOT	Hidden Policy	Decryption Outsourcing	Cloud/Fog Outsourcing	Collaboration	Technique
Feng et al [52]	✓	✗	✓	Cloud	✗	Generic parallel outsourced decryption based on Spark and MapReduce
Chen et al [67]	✗	✗	✗	✗	✓	Collaboration by using integrated access trees
SHARE-ABE	✓	✓	✓	Fog	✓	Decryption outsourcing & Hidden access policy using false attributes & Collaboration in the same group by using collaboration attributes

Existing ABE approaches that support collaboration are limited in that (1) The decryption computation cost increases proportionally with the complexity of the access structures, (2) The privacy of the access policy is not guaranteed. In GO-ABE and in [67], the decryption process is energy-intensive since all operations are executed on the user's device. In addition, the access policy is sent in clear text. Furthermore, the data owner cannot control the collaboration since users can share all their attributes without

exception. Similarly, the solution in [66] is resource-intensive since it does not use outsourcing. Also, adding a translation value increases the computation cost. Moreover, the access policy privacy is not supported. To the best of our knowledge, our work is the first that is resources-efficient, supports policy privacy, and allows an efficient and controlled collaboration for users in the same group. Finally, our approach is more relevant as it uses Fog computing for decryption outsourcing.

3.5 Conclusion

In this chapter, we discussed how Attribute-based Encryption gives greater fine-grained access control. We demonstrated that despite these advantages, the implementation of Attribute-based Encryption in resources-constrained devices carries many challenges. We primarily focused on three challenges: resource limitations, hidden access policy, and user collaboration. Then we reviewed existing solutions aiming to overcome these challenges more specifically in the context of performance requirements, access policy privacy, and user collaboration and we identified their shortcomings. In the following chapters, we will discuss our proposed solutions for collaborative data sharing based on ABE. In the first contribution, we focus on outsourcing the decryption process to the Fog nodes and hiding the access policy. The second contribution takes into consideration an addiction aspect which is the collaboration in the same group.

Part II

Contributions

Chapter 4

A Multi-Fog and Privacy-Preserving Data Sharing Scheme for Resource-Constrained Devices

4.1 Introduction

Nowadays, devices range from mobile phones and laptops, to desktops and servers. Many devices have limited computing power and energy, which makes the development of cryptographic algorithms such as ABE schemes more challenging. Indeed, as discussed in the previous chapters, ABE approaches suffer from high complexity and overhead, making them inefficient for resource-restricted devices.
In this chapter, we describe our solution to adapt the ABE scheme to resources-constrained devices. The basic idea is to use Fog nodes collaboration and a new partial decryption approach with a hidden access policy to achieve low computation overhead in addition to secure and fine access control.

Our architecture, model and assumptions are presented in section 4.2.1. and 4.2.2. Security Analysis and Performance evaluation are detailed respectively in sections 4.3 and

4.4. In Section 4.5 of this section, we describe an application scenario of Our schema. Finally, Section 4.6 concludes this chapter.

4.2 The Proposed Approach

In this section, we describe our proposed architecture adapted to resource-constrained devices (Figure 4.1). It is scalable and able to store the large amount of data. It represents a new scheme based on the CP-ABE algorithm and Fog computing.

It consists of five parties: Trusted Authority (TA), Data Owner (DO), Data User, Fogs, and the Cloud. The TA is responsible for system initialization, authenticating the users' attributes, creating and sending the secret keys to the users, and generating intermediaries' keys to the Fogs. The Data Owner is the user who wants to upload and share his data; it is also his role to specify the access policy, which is used to encrypt the data. The policy is used to control who can access this shared data. The data user is the one who wants to access the shared data; he solicits the TA by sending his attributes in order to obtain a private key that will be used to decrypt the data. The Cloud provides a storage service to users to access the shared data anywhere and anytime. Fogs are entities that collaborate and help users partially decrypt the data.

In our scheme, we use CP-ABE to encrypt data and achieve fine access control before storing them on the Cloud. However, use CP-ABE in resources-constrained devices (sensor devices) is a real challenge. In CP-ABE, the computational cost during the encryption and decryption phases increase exponentially with the access policy's complexity. This is a considerable limitation when devices are limited in terms of resources (CPU, energy, etc.). Another drawback of ABE is that the access policy is sent in clear text along with the cipher-text. A malicious user can obtain both the cipher-text and the associated access policy. The latter contains some sensitive information (like social security number, name, etc.) that can be exploited to compromise the legitimate user's privacy.

To tackle the first challenge that is the heavy computation cost in the decryption process, we propose using Fogs approaches to the outsourced heavy decryption process. Our proposition inherits attractive properties from Fogs such as being closer to the user, and the deployment of such devices is cheaper, more resource-efficient, and network efficient. This potentially offers users low-latency-guaranteed applications. In our idea, the Fogs are used to help the end-user decrypt the data. These last works together to reduce the bandwidth and partially decrypt the data. Note that each Fog decrypts data partially according to attributes that it manages. The TA (Trusted Authority) creates intermediate keys for the Fog nodes using the user's secret key to delegate the decryption operation. The Fog uses this intermediate key to decrypt data partially . This means that the computational decryption complexity of the resources-contained devices is independent of the number of attributes.

To tackle the second challenge, which is to hide the access policy. We propose to add false attributes to hide the access policy. The Trusted Authority divides the set of attributes overall available Fogs so that each Fog manages its own set of attributes. When the user (Data Owner) creates an access policy, he divides the access policy according to the attributes that each Fog manages and adds false attributes to each access policy's subtree so that Fogs nodes cannot deduce the real attributes. Also, Fog nodes cannot deduce the valid attributes of users in the decryption process, even if the Fog nodes are compromised or collude. This operation is performed by taking into account the number of available Fog and according to the set of attributes managed by the Fog node. In this way, the Fog nodes will not be able to deduce which attributes participated in the decryption phase.

4.2.1 Models and Security requirements

In this section, we present a system model for sharing data on a Cloud server by outsourcing encryption costs on Fog nodes. We review the security assumption and requirements.

Then we present the different phases and algorithms of our approach.

4.2.1.1 System Model

Figure 4.1 illustrates a Fog architecture for secure data sharing. It relies on the following entities, permitting a user to store the data securely, and share it with multiple users that use a constrained -devices:

- The Trusted Authority (TA): The administrator of the entire system. It is responsible for the system initialization, user's attributes management, creating and sending secret keys to users, and generating intermediate keys for the Fogs.
- Data owner: The entity that outsources its data to Cloud servers. The data owner specifies the access policy used to encrypt the data (Cipher-text) before uploading it to the Cloud server. This access policy represents the access structure in tree form, which contains the attributes.
- Cloud service provider: Its role is to provide a public platform for the user to store and publicly share his data. This entity does not provide control access to the encrypted data.
- Fogs: The Fogs are entities that collaborate and help users partially decrypt the data. In our scheme, each Fog is responsible for a set of attributes and only decrypts the set of attributes they manage.
- consumers (users): The end-users. A user can decrypt the ciphertext only if his attributes satisfy the access policy, integrated with the encrypted data.

4.2.1.2 Security Assumption and Requirements

In our proposed approach, we assume that TA is a trusted entity as in any system that uses ABE. We also assume that Cloud and Fogs are semi-trusted entities, ie., the Cloud

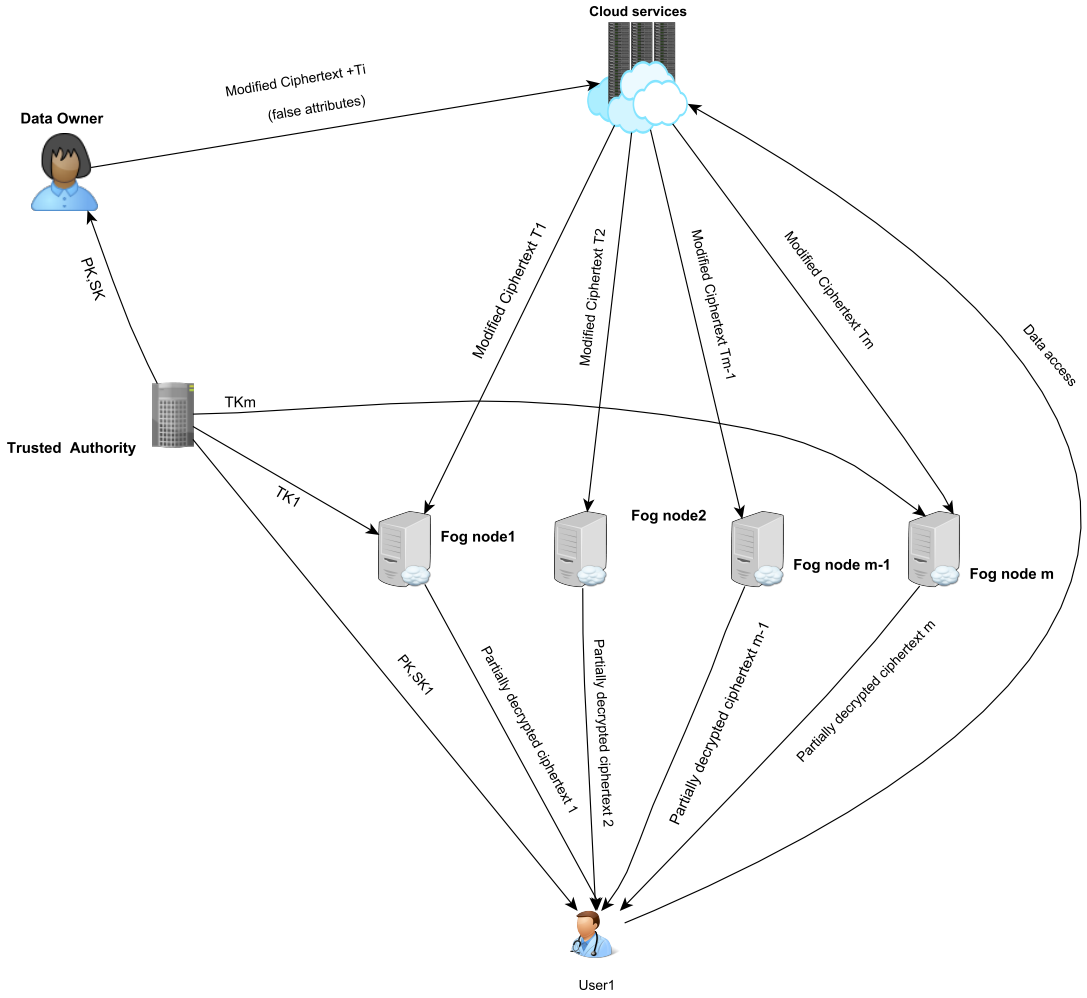

Figure 4.1: Scheme of the proposed solution[47]

and the Fogs apply the protocols but curious entities. Also, we suppose that each Fog manages a set of attributes in such a way that: $\forall i \neq jN_i \cap N_j = \emptyset$ where N_i, is the set of attributes belonging to the Fog_i.The communication between the entities is secure. The security model of our scheme covers the following aspects:

- Data confidentiality: The content of the data must be confidential. Users not defined as recipients by the data owner should not be able to access data, including Cloud, Fogs, and end-users (consumer).

- User collision resistance: Users cannot combine their attributes to satisfy the access policy.
- Fine-grained access control: The encrypted data can only be accessed by users whose attributes satisfy the access policy defined by the data owner. Such access policy must be flexible and expressive.
- Hidden access policy: In the access policy, the attributes should be hidden so that the users cannot deduce sensitive information.

4.2.2 Different Phases and Algorithms Description

This section describes the various phases and algorithms of our approach. The attributes universe is known by all entities in the system, we use the notation Att to represent attributes in the access policy.

4.2.2.1 Initialization phase

In this phase, the Trusted Authority generates two keys, a public key (PK) that is shared for all entities in the system and a Master Key (MK) that will be kept secretly. After creating the keys, the TA assigns each user its own attributes. At the end of this step, each user will know the public key and the sets of all the attributes in the system.

- **Setup Algorithm**

  - **Setup** (τ, N, f): The algorithm takes as input a security parameter τ, the set of universal attributes N and the number of available Fogs f. The algorithm chooses a bilinear group G with an order $O = p_1p_2p_3p_4$, for each attribute $A_i \in N(1 \leqslant i \leqslant n)$ where n is the number of attributes in the universe N. Then is selects $h_i \in {Z_N}^*$ and finally selects a random element $\alpha, \beta \in Z_N^*$ and

$g \in G_{p_1}$. The public key is defined by:

$$PK = \{N, g, y = (g, g), L = g, H_i = g^{h_i} (1 \leqslant i \leqslant n)\}$$

and the master key by:

$$MK = \{\alpha, \beta\}.$$

- The algorithm divides the set N by the number of available Fogs f. This means $N = N_1 \cup N_2 \cup, \ldots, \cup N_f$ in such a way $\forall i \neq j, N_i \cap N_j = \emptyset$ where N_i is the set of attributes belonging to the Fog_i. This phase is executed by the Trusted Authority (noted TA).
- When the user requests his private key with his set of attributes. The TA chooses a random variable θ that will be the private key of the user ($SK = \theta$).

4.2.2.2 Encryption phase

When the Data Owner wants to share information with another user in the system according to an access policy, he creates an access policy T in tree form. He divides This tree into several subtrees T_i according to the available Fogs in the system and according to the attributes and their belonging to the Fogs (The TA sends the list of available Fogs and their attributes) (Figure 4.1). After obtaining the subtree T_i, the DO (Data owner) adds false attributes that belong to the destination Fogs for hiding the real attributes. He chooses random numbers $\{s_1 \ldots s_f\}$ corresponding to each Fog $\{Fog_1 \ldots Fog_f\}$ where s_i is shared by all the attributes in T_i. Each s_i is shared for each node of the access tree T_i. The secret s_i is divided according to the "Top-Down approach" that means the secret s_i is divided by $(t - n)$ Shamir secret sharing approach from the root to leaf node when n is number of all child node and t is number of child node for recover secret s_i. Each real attributes in T_i will contain the share λ_i of s_i. In contrast ,the false attributes will not

contain the share λ_i of s_i , moreover, the false attributes will be eliminated in the partial decryption phase. After that, the DO sends the ciphertext with all T_i to the Cloud for storage.

- **Encryption Algorithm**

  - In this phase, the *DO* executes the Encryption primitive denoted **Encryption** (PK, M, T, L) as follows:

    The Encryption algorithm takes as input the public key PK , the message M and the access policy T in the tree form and L which represents the list of available Fogs with their attributes (N_i).

  - The algorithm divides the tree T into several subtrees T_i according to the number of available Fogs. Each subtree will contain the attributes of the destination Fog.

  - The Sender adds false attributes to the subtrees according to the universe of attributes of the destination Fog. Let U_i be the set of attributes of the subtree T_i after adding false attributes. In the subsequent step, The Sender chooses a random numbers $\{s_1 \dots s_f\}$ corresponding to each Fog $\{Fog_1 \dots Fog_f\}$ where s_i is shared by all the attributes in T_i.

  - The algorithm shares the secret s_i as follows: a polynomial $q_i(x)$ degree $k_i - 1$ is chosen for each node (including the leaf node) in T_i where $k_i = |T_i|$ (number of elements in (T_i). These polynomials are generated in a recursive manner starting from the root node r. We define $q_{ir}(0) = s_i$ (where r represent the root node in the tree) then other value $k_i - 1$ are defined randomly to complete the construction. Once all the polynomials have been defined, we put $\lambda_{xi} = q_{xi}(0)$ for each node x in T_i, we choose random elements $Z_0, \{Z_i\}_{A_i \in U_i} \in G_{p4}$ knowing that $att(x) = A_i$ and $index(y)$ is attributes index of y in T_i , the ciphertext is

generated as follows:

$$CT = \{E = My^s, E_0 = g^s Z_0, CT_i\}$$

$$CT_i = \left\{ \begin{array}{l} \forall A_i \in T_i : E_i = L^{\lambda_{xi}} H_i^{s_i} Z_i \\ \\ \forall A_i \notin T_i : E_i = H_i^{s_i} Z_i \end{array} , T_i \right\}$$

Where $s = \sum s_i$.

The ciphertext is formed as CT include CT_i that is stored in the Cloud.

4.2.2.3 Decryption phase

This phase contains two phases: partial decryption and final decryption. In the partial decryption, When a user wants to access the shared data, he requests his private key from TA with his attributes (S). The TA chooses two random variable θ and t, where ($SK = \theta$) will be the private key of the user and t it used with the set of user's attributes to create the transformation keys TK_i for each available Fog_i. The Fogs can use TK to decrypt the data partially. Both keys are sent securely. The partial decryption at the level of Fog_i is performed with the TK_i key. Each Fog_i decrypts the data partially without knowing which attribute participated in the partial decryption. Each Fog sends partially decrypted data to the user to recover the message M. Finally, in the final phase and After the user receives all partially decrypted data, he recovers the message with his private key SK.

- **Decryption Algorithm**
  - When a user wants to access the shared data, he sends a request to the Cloud about the encrypted data and requests the TA to create the transformation keys TK.

So, the TA executes the primitive **KeyGen**(PK, MK, S, θ, f) as follows:

The TA (Trusted authority) creates the transformation keys TK for each Fog. For that, the TA starts the key generation procedure where this key makes it possible to perform a partial decryption. To create TK **KeyGen** chooses a random element $t \in Z_N^*$ and $R, R_0, \{R_i\}_{A_i \in s_i} \in G_{p3}$, then returns the transformation key for each Fog.

Formally:

$$TK = \{D = g^{(\alpha-\beta t)\theta} R, D_0 = g^t R_0, \forall A_i \in S_i : D_i = H_i^t R_i\}$$

Finally, the TA distributes the TK_i key to Fog_i.

– Upon receipt of the TK_i key and CT_i , Fog_i. executes the following function:

DecrypPartial(CT_i, TK_i): This algorithm takes as input CT_i and TK_i. When the Fog_i receives CT_i it uses its transformation key TK_i to partially decrypt the ciphertext. Two recursive functions are used:

DecryptNode_$CT_i(CT_i, x)$ which takes as input CT_i and the node x which belongs to T_i.

DecryptNode_$TK_i(TK_i, x)$ which takes as input the transformation key and the x node.

The algorithm of **DecryptNode_**CT_i and **DecryptNode_**TK_i is defined by the following instructions:

If the node x is a leaf node

Set

$DecryptNode\_CT_i(CT_i, x) =$

$$E_i = \begin{cases} {}^{\lambda_{xi}} H_i^{s_i} Z_i & if A_i \in T_i \\ {H_i}^{s_i} Z_i & if A_i \notin T_i \end{cases}$$

$DecryptNode\_TK_i(TK_i, x) = D_i = H_i^t R_i$

We consider the case where x is an internal node. The two functions $DecryptNode\_CT_i$, and $DecryptNode\_TK_i$ are executed in the following way: (knowing that the direction of execution is root to down) For each node y that is the child of x $DecryptNode\_CT_i$ and $DecryptNode\_TK_i$ are invoked. The result is saved respectively in F_y and K_y, let Q_x a set of y nodes child that belongs to T_i and $Q_{x'}$ the set of y nodes that does not belong to T_i. We have $Q_x \cup Q_{x'}$ = all the children of the x in the T_i tree. If y is a node then we calculate:

$F_x = \prod_{y \in Q_x \cup Q'_x} F^{l_y v_x(0)}$
$= \prod_{y \in Q_x} (L^{\lambda_{x_i}} H_i^{s_i} Z_i)^{l_y v_{x(0)}} \cdot \prod_{y \in Q_x \cup Q'_x} (H_i^{s_i} Z_i)^{l_y v_{x(0)}}$
$= \prod_{y \in Q_x} g^{\beta \lambda_{y_i} . l_y v_x(0)} \cdot \prod_{y \in Q_x \cup Q'_x} H_i^{s_i . l_y v_x(0)} \cdot \prod_{y \in Q_x \cup Q'_x} Z_i^{l_y v_x(0)}$
$= g^{\beta \lambda_{x_i}} F_{x,1} F_{x,2}$

And

$K_x = \prod_{y \in Q_x \cup Q'_x} . K^{l_y v_x(0)}$
$= \prod_{y \in Q_x \cup Q'_x} H_i^{t . l_y v(0)} \cdot \prod_{y \in Q_x \cup Q'_x} R_i^{l_y v_{x(0)}}$
$= K_{x,1} K_{x,2}$

If the node is non-leaf node we calculate:

$$F_x = \prod_{y \in Q_x \cup Q_{x'}} .(g^{\beta \lambda_{yi}} F_{x,1} F_{x,2})^{l_y v_{x(0)}}$$
$$= \prod_{y \in Q_x} (g)^{\beta \lambda_{yi} . l_y v_x(0)} . \prod_{y \in Q_x \cup Q'_x} F_{y,1}^{l_y v_{x(0)}} . \prod_{y \in Q_x \cup Q'_x} F_{y,2}^{l_y v_{x(0)}} = g^{k \lambda_{xi}} F_{x,1} F_{x,2}$$

And

$$K_x = \prod_{y \in Q_x \cup Q'_x} .K^{l_y v_{x(0)}}$$
$$= \prod_{y \in Q_x \cup Q_{x'}} .K_{y,1}^{l_y v_{x(0)}} . \prod_{y \in Q_x \cup Q'_x} .K_{y,2}^{l_y v_{x(0)}}$$
$$= K_{x,1} . K_{x,2}$$

In the previous equation, we have $F_{x,1} = K_{x,1}$, the parameter $v_x = \{index(y)/y \in Q_x \cup Q_{x'}\}$ and $l_y v_{x(0)}$ is the coefficient of lagrange. If we call both functions from root r of T_i then we obtain:

$$A = DecryptNode\_CT_i(CT_i, r) = g^{ks_i} . F_{r,1} . F_{r,2}$$

And

$$B = DecryptNode\_TK_i(TK_i, r) = K_{r,1} . K_{r,2}$$

We calculate :

$$C_i = e(A, D_0)/e(E_0, B)$$
$$= e(g^{\beta s_i} . F_{r,1} . F_{r,2}, g^t R_0)/e(g^{s_i} Z_0, K_{r,1} . K_{r,2})$$
$$= e(g^{\beta s_i}, g^t) . e(F_{r,1}, g^t) . e(F_{r,2}, g^t) . e(g^{\beta s_i} . F_{r,1} . F_{r,2}, R_0)$$

$/e(g^{s_i}, K_{r,1}).e(g^{s_i}, K_{r,2}).e(Z_0, K_{r,1}.K_{r,2})$

$C_i = e(g,g)^{\beta t s_i}$.

And also :

$P_i = e(E_0, D)$

$= e(g^{s_i} Z_0, g^{(\alpha-\beta t)\theta} R)$

$= e(g^{s_i}, g^{(\alpha-\beta t)\theta} R)$

$= e(g,g)^{s_i(\alpha-\beta t)\theta}$

- Finally, the Fog sends the partial decryption C_i and P_i to the user.
- Upon receipt of all shares parts of Fog_i, the user executes the following function: **Decryption**(C_i, P_i, SK, E): This algorithm is executed by the user. If the user receives all the parts which are partially decrypted from the Fog, then he knows that his attributes satisfy the access policy. Otherwise, he rejects the decryption. When the user receives all the partially decrypted parts, he uses his private key $Sk = \theta$ and the ciphertext transformed by the Fog (C_i, P_i) to recover the original message.

  Formally:

$$\frac{E}{(\prod P_i)^{(\frac{1}{\theta})}.\prod C_i} = \frac{E}{(\prod (e(g,g))^{s_i(\alpha-\beta t)})^{(\frac{1}{\theta})}.\prod e(g,g)^{\beta t s_i}}$$

$$= \frac{E}{(e(g,g)^{(\alpha-\beta t)\theta \sum s_i})^{(\frac{1}{\theta})}.e(g,g)^{\beta t \sum s_i}}$$

$$\frac{E}{(e(g,g)^{s(\alpha-\beta t)\theta})^{(\frac{1}{\theta})} e(g,g)^{s\beta}} = \frac{E}{e(g,g)^{s(\alpha-\beta t)}.e(g,g)^{st\beta}}$$

$$\frac{E}{e(g,g)^{s\alpha}} = \frac{Me(g,g)^{s\alpha}}{e(g,g)^{s\alpha}} = M$$

4.3 Security Analysis

In this section, we discuss the security properties of the proposed approach.

- **Data Confidentiality:** The confidentiality requires that the Cloud and the Fog cannot learn the encrypted data, In the decryption-outsourcing algorithm, the Cloud is responsible only for storing the encrypted data as $Me(g,g)^s$ where s is kept secret by the user. While, the Fogs are responsible only for the partial decryption of the data, and since the transformation keys TK_i are generated by TA with the secret key of the user, only the end user where his attributes correspond to the access policy, can recover the encrypted data, in other words, Fogs cannot recover random value s where this value is divided among the Fogs in the encryption process even if the Fogs cooperate with each other, since in the processing of the partial decryption the s_i are blinded with the secret key of the user $SK = \theta$. Thus, we conclude that our scheme is secure in protecting the confidentiality of the message.

- **Collusion resistant:** the collision resistance is the property that the CP-ABE assumes. In our solution, the algorithm Keygen generates a different random values t for each user and which is integrated into the key of transformation. It means that each key of the user is randomized, this means that users can not combine their keys to decrypt the data, so malicious users can not collaborate to expand their access privileges including Fog nodes since the transformation key contain the random value t .

- **Hidden access policy:** in our scheme the DO adds false attributes to the access policy, with this method the malicious users and even the Fogs can not have the real attributes even if the Fogs cooperate with each other, this will lead to the

addition of several false attributes which further complicates the task of having the right attributes. Also, the Fogs and even the users cannot know which attribute participated in the decryption of the data as all the attributes of the users whether they belong to the access policy are being applied in the decryption process.

4.4 Performance Evaluation

In this section, we present the analysis of the performances of our proposed scheme. We start by giving the experiment setting. Then, we compare our solution to the related works in terms of overhead and execution time by using Charm framework [70].

4.4.1 Computation Cost

In this part, we present the computation complexities of our scheme. We are interested in the computation cost related to the execution of the decryption algorithms performed by the user (U) and the Fogs nodes. This choice is motivated by the fact that we used partial decryption that is delegated to Fogs. In the next paragraphs, we use the following notations:

- E: exponentiation in $G1$.
- P: computation of a pairing function e.
- n: is the number of attributes used in the access policy
- nr_i: is the number of real attributes in the access policy and managed by Fog_i
- nf_i: is the number of false attributes in the access policy and managed by Fog_i
- nc_i: is the number of collaboration attributes managed by Fog_i

- nl: is the number of non-leaf nodes when computing the secret of the root node from leaf nodes in access policies.
- Ln_i: is the number of non leaf node satisfied by the Fog_i
- ny: is the number of combined attribute set Y.where y set contain all attributes of the same group of the user.
- N_f: is the number of Fog nodes involved in the decryption process.
- k: is the number of attributes associated with the private key of a user.
- $|\alpha|$: expresses the number of LPNodes in the access tree

The simple multiplication operation is not taken into account because it is negligible compared to the pairing and exponentiation operations.

In the following sections, we compare our work and the different approaches both at the End-user and Fog levels.

4.4.1.1 At the End-user Level

End-user computations are depicted in table 4.1. Traditional CP-ABE and the approaches proposed in [66] and [62] execute the full ABE algorithm at the end-user level. The decryption computation cost in the latter approaches are given by $(2n+1)p+(n+nl)E$, $(2ny+nc+2)p+(ny+nl)E$ and $3P+2(n)E$ respectively. The precedent approaches complexities show that the decryption computations are very resource-intensive, inefficient, and are not adapted to resource-constrained devices. In [61; 69],the authors proposed an outsourced ABE scheme for Fog computing applications where the computation consists of only one exponentiation (1E). However, the drawback of these approaches are as follows:

- [69], the approach does not support hidden access policy.
- In [61], support the hidden access policy. Nevertheless, In this approach, the access policy is specified based on Linear Secret Sharing Scheme(LSSS) structure (in matrices form), and the decryption process is slow. By contrast, in our collaborative approach, the access policy is specified based on the Tree structure, which is as expressive as LSSS. Our method additionally has the advantage of being quicker and more efficient when it comes to decryption.

Table 4.1: Computation Costs at the End-user Level

Scheme	**Complexity of the Decryption Process**
CP-ABE	$(2n+1)p+(n+nl)E$
Wang et al [62]	$3P+2(n)E$
Belguith et al [61]	$(1E)$
Xue et al [66]	$(2ny+nc+2)p+(ny+nl)E$
Zhao et al [64]	$(2E)$
Tu et al [69]	$(1E)$
Collaborative approach	$(1E)$

4.4.1.2 At the Fog-nodes Level

In the Collaborative scheme, the Fog nodes are charged with expensive computation operations. Also, the computation on the Fog node level is reduced by introducing the collaboration between Fogs , where the set of attributes is assigned to each Fog. Table 4.2 shows the decryption computation cost of the proposed approaches at the Fog node level (which is given by $(2nr_i+2nf_i+nc_i)p+3|Ln_i|E)$ is better than [64]. Table 4.3 also shows that our collaboration approach(Fog in the parallel scheme) generates many interactions between Fogs and end-users (which is given by 2x(Nf + 1)), which results in a growing expense of networking and energy use.

Table 4.2: Computation Costs at the third party server Level

Scheme	Complexity of the Decryption Process
Tu et al [69]	$(2+2n)P+E$
Zhao et al [64]	$(k\|\alpha\|+\|\alpha\|)P+(k\|\alpha\|+\|\alpha\|)E+P+E$
Collaborative approach (Fog)	$(2nr_i+2nf_i+nc_i)p+3\|Ln_i\|E$

Table 4.3: Number of Communications between the End-user and Entities in the Decryption Process

Scheme	Number of Communications
Collaborative approach	$(2\times(N_f+1))$

4.4.2 Evaluation

We used Charm [70] to implement our scheme and the related approaches. Charm is a framework based on phyton that facilitates the rapid prototyping of cryptographic schemes and protocols. The experiments are conducted on a machine with a Xeon E5-2630 (2,20GHz) processor and 32 Go RAM. We used a Virtual Machine with one core and 2.00 GB of RAM. The VM runs the Ubuntu16.04 64-bit OS with python 2.7. The number of attributes used in the experiments are $L = \{1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 100\}$. In the experiments, we evaluate the time of one multiplication, one pairing, and one exponentiation. The time of the multiplication operation is very short, thus it can be ignored in the analysis. The time of a single pairing operation was evaluated to 1.5 ms, and the time of one exponentiation operation is on average 1.8 ms.

In our experiments, we evaluate the time cost of data decryption and compare it with the related approaches including: traditional CP-ABE, Wang et al [62], Belguith et al [61], Xue et al [66], Zhao et al [64] and Tu et al [69] .

Figure 4.2 shows the decryption computation time on the user's sides with respect to the number of attributes in the access policy. The figure shows that in CP-ABE, Wang

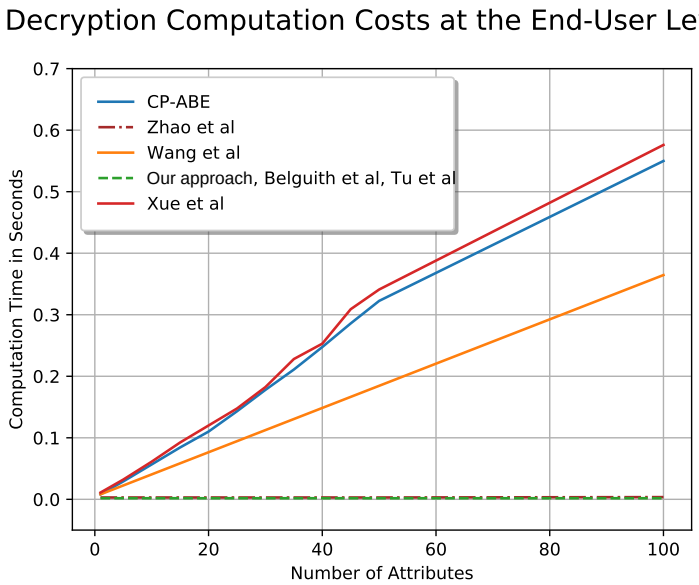

Figure 4.2: Comparison of Decryption Computation costs at the End-User level[9]

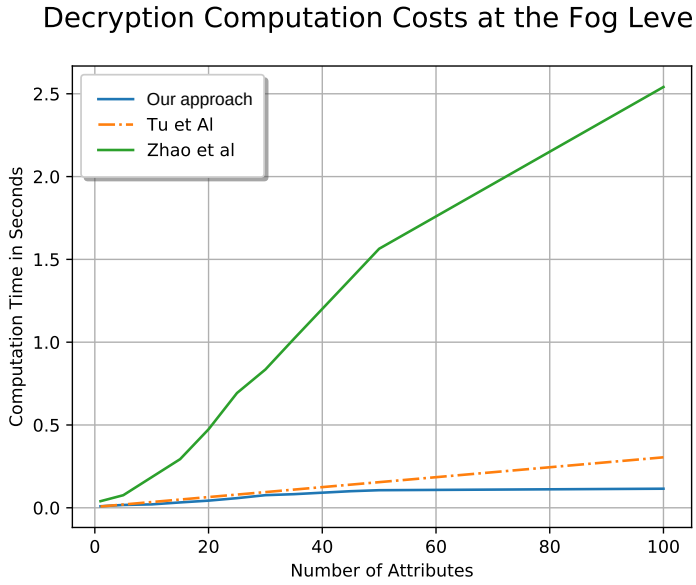

Figure 4.3: Decryption Computation Costs at the Fog Level[9]

et al [62] and Xue et al [66], the decryption time on data owners grows as the number of attributes in access policy increases, as opposed to Belguith et al [61], Tu et al [69], Zaho et al [64] and Our proposed approach, where the decryption times do not depend on the number of attributes. This is because the decryption operations are outsourced.

We see that in Belguith et al [61], Tu et al [69] and our approach illustrate findings that are identical and are significantly lower than Zaho et al [64]

We also evaluate the decryption time on the Fog side. The results are shown in Figure 4.3. In the experiments, we use three Fog nodes for our approach. We compare our work to approaches that use Fog outsourcing, i.e Tu et al [69] and Zhao et al[64]. We observe that the experimental results in these approaches including the proposed approach follow a linear relationship (approximately) as the number of attributes increases. In Figure 4.3, it is clear that our approach gives the best results. The explanation for this is that we partition the attributes through many Fogs where each Fog node handles a different subset of attributes, and also, each Fog treats just a limited portion of the access tree.

4.5 Application Scenarios

In this section, we introduce an application scenario. Our schema can be used in healthcare systems, where wearable devices can detect and collect user's health data. The system is composed of entities such as medical insurance, analysis laboratory, private hospitals, hospitals, where each entity manages a set of attributes. Also, each entity is connected to a Fog that will manage these attributes. A doctor or a member of the patient's family is authorized to decrypt the data (according to the access policy). Because the doctor or the family member uses constrained devices, they request the Fog to decrypt the data partially. According to our proposed method, the scenario is defined as follows:

1. After receiving the collected data on his smartphone, the patient (or data owner) de-

fines -by utilizing an application GUI, for Example- the access policy which specifies who can access the data. For Example, a doctor.

2. Then, the device splits the access policy by considering attributes that are managed by each Fog. Next, it pads them with false attributes and sends each part to the corresponding Fog.

3. The data file is encrypted and sent to the Cloud along with the complete access policy with is also padded with false attributes.

4. When a doctor wants to read the data file, he connects to the Cloud to get it. His attributes are sent to the Trusted Authority, which will create the intermediate key.

5. This intermediate key is sent to the Fog nodes, which partially decrypt the ciphertext. This process also includes testing the partial access policy (see CP-ABE section)

6. After decryption, all Fog nodes send the partial ciphertext to the doctor, which decrypts the complete ciphertext using his private key.

4.6 Conclusion

In this chapter, a new collaborative approach based on CP-ABE is proposed. We used Fog nodes to reduce the bandwidth and decrease the decryption cost by delegating the decryption process to the Fog nodes. The Fog decrypts data partially according to attributes it manages by employing the intermediate key created by the TA (Trusted Authority) using the user's secret key. The proposed solution also preserves the privacy of the access policy so that the sensitive information in the access policy is not disclosed. This is performed by introducing false attributes that are mixed with the real attributes. The Data Owner introduces these false attributes after dividing the access policy according to the

attributes that each Fog manages. This operation is performed by considering the number of available Fog and according to the set of attributes managed by the Fog node. The experiments also show that the proposed approach outperforms state-of-the-art approaches and gives promising results.

Chapter 5

SHARE-ABE: Collaborative Encryption Based on Group-Oriented Attributes in Chained Multi-Fog Scheme

5.1 Introduction

Despite the numerous benefits of Cloud computing, there are still many challenging concerns that prevent Cloud computing from being generally adopted. The One of the most important challenges is the privacy and data security for users. As a result, safe access control has become a complex problem in public Cloud storage, where specific security measures cannot be implemented directly. Many studies have been conducted in recent years on data access control in public cloud storage. Ciphertext-policy Attribute-based Encryption (CP-ABE) is considered one of the most suited schemes since it can ensure data owners' direct control over their data while also providing a fine-grained access control service. Nonetheless, these approaches suffer from their overhead on constrained

devices, and the confidentiality of the access policy is not guaranteed. Therefore, in the previous chapter, we have proposed an architecture based on the CP-ABE algorithm to guarantee fine access control and confidentiality of the data collected. We have addressed the problem of the cost of decryption on devices with limited resources by delegating the operation decryption at fog nodes without revealing the original message. We have also added false attributes to hide the access policy to ensure the confidentiality of the access policy. But, present CP-ABE methods can only provide access to persons who hold attribute sets that fulfill the access policy. However, in many cases, the secret knowledge cannot be retrieved alone by a single person. In this chapter, we present our second contribution, SHARE-ABE: Collaborative Encryption Based on Group-Oriented Attributes in Chained Multi-Fog Scheme. In The first part, we describe the motivation in Section 5.2. Model and Security requirements are presented in Section 5.4. In the second part, we present an overview of our proposed scheme. Then, we provide a detailed description of the different phases of our approach. We also present the Security Analysis and Performance evaluation respectively in sections 5.5 and 5.6. Finally, Section 5.7 concludes this chapter.

5.2 Motivating Scenario

Figure 5.1 depicts an example of an e-health architecture based on Fog computing. Sensor devices send the encrypted patient data to Cloud storage. Then, medical staff that have the right access attributes can access the data when needed by using smartphone Tablets and PCs. One of the main benefits of using Fog computing is to offload both medical Sensors and Access devices (i.e, smartphones, Tablets, and Pcs) from heavy computations and also to make network communications more efficient.

Let's take the example of the access policy depicted in Figure 5.2. A Hospital has multiple departments and, each one can have multiple profiles: Professor of Medicine

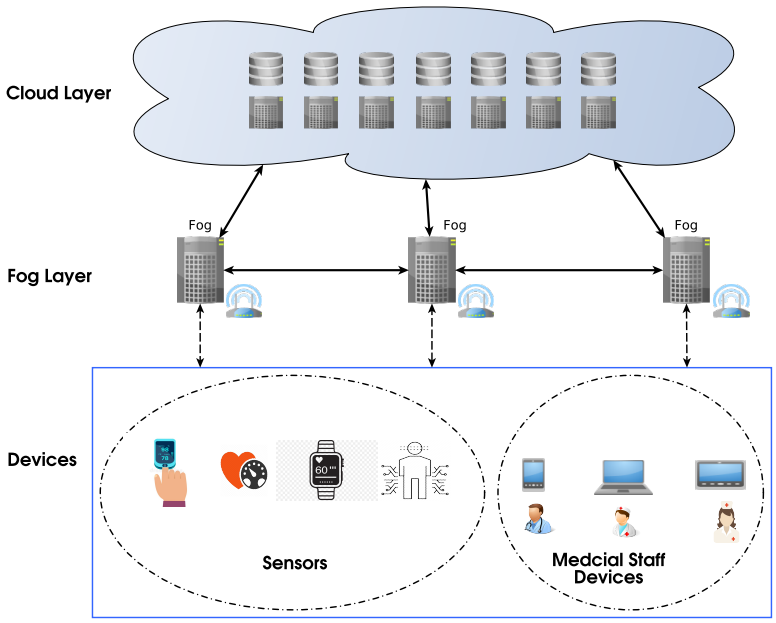

Figure 5.1: A Typical e-healht Architecture[9]

(Pr), Master Assistant in Medicine (MA), and Doctor of Medicine (Do). Data concerning patients (collected from sensors for example) is encrypted and stored in the Cloud. We want to allow practitioners of rank professor in the cardiology department (Cardio_dept) to decrypt the data. In some situations -like in emergencies-, we also want to allow practitioners of rank Do or MA to have access to the data file.

This is done by allowing users to collaborate by sharing their attributes (inside the same group, for example in the same department).

In the example, some users have attributed the {access} attribute, which is used for collaboration.

However, we can see that if Do and Ma can share the attributes {Ma} and {Do} respectively, they can satisfy the access policy even without having the {access} attribute.

As a solution, the collaboration must be only on allowed the attribute {access}, which prevents attribute collaboration in a malicious way.

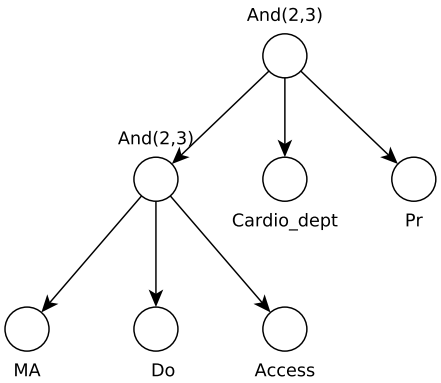

Figure 5.2: Example of an Access Policy

Motivated by the above observations, we address the following challenges:

- How to overcome the decryption overhead on a resource-constrained device,
- How to protect the confidentiality of the access policy so that the malicious user cannot exploit sensitive information in the access policy,
- How to achieve a controlled collaboration in the same user's group to satisfy the access policy and to access the data,
- How to limit the communication between the user (consumer) and the system's entities to conserve the energy and network communications in the resource-constrained environment.

In this chapter, we propose a new approach: SHARE-ABE which can be summarized as follows:

1. To the best of our knowledge, our work is the first to address user collaboration in the same group and provides both data confidentiality and collaborative access control. In the proposed approach, users can have different sets of attribute. In

addition, a hidden access policy is used, and the decryption process is outsourced to Fog nodes.

2. We introduce a new construction for collaboration attributes, which are included in the access policy. These attributes are defined by the data owner to control collaboration among users in the same group.

3. We use a proxy to add false attributes to the access policy, so that malicious users cannot guess the real attributes. We add this proxy in order to reduce the cost overhead at the level of the owner of the data.

4. We use a new chain decryption scheme to conserve the energy of the resource-constrained devices. Each Fog decrypts the access policy based on the attributes it manages and passes the results to the next until the user receives the decrypted message.

5. We introduce a new function $R(x)$, which can retrieve the list of users that can collaborate in the same group, when collaborative nodes are used in the access policy.

6. We thoroughly analyze the security properties and evaluate the performance of our proposed scheme.

5.3 The Proposed Approach

In this section, we give an overview of our proposed scheme; then we provide a detailed description of the different phases of our approach.

5.3.1 Models and Security requirements

The system model and the security requirements that we consider in the development of our solution are as fellows:

5.3.1.1 System Model

In our scheme, we focus on securing data sharing and the collaboration between users of the same group. The system model of our work is depicted in Figure 5.3. It consists of six entities: trusted entity, data owner, data consumers, Cloud service provider, proxy, and Fogs.

- **The Trusted Authority (TA):** The administrator of the entire system. It is responsible for the system initialization, user's attributes management, creating and sending secret keys to users, and generating intermediate keys for the Fogs.
- **Data owner:** The entity that outsources its data to Cloud servers. The data owner specifies the access policy, which is used to encrypt the data (Ciphertext) before it is uploaded to the Cloud server. This access policy represents the access structure in tree form, which contains the attributes.
- **Proxy:** Its role is to reduce the cost overhead at the level of the owner of the data. It is used to hide the access policy by adding false attributes in order to hide the real attributes.
- **Cloud service provider:** Its role is to provide a public platform to the user where he can store then publicly share his data. This entity does not provide control access to the encrypted data.
- **Fogs:** The Fogs are entities that collaborate and help users to partially decrypt the data. In our scheme, each Fog is responsible for a set of attributes and only decrypts the set of attributes they manage.

- **Data consumers (users):** The end-users. A user can decrypt the ciphertext only if they fulfill one of the following conditions:

    1. His set of attributes satisfy the access policy, which is integrated in the encrypted data,
    2. If the access policy allows collaboration, he can collaborate with other users in order to satisfy the access policy. The collaboration attributes are defined in the access policy by the data owner.

The Data Owner creates an access structure defining the attributes of the users allowed to decrypt the data. Then, he creates a ciphertext using the public parameters (PK), the master key (MSK) and access structure. The ciphertext is transmitted to the proxy which adds false attributes to the access policy to obfuscate the real attributes. It is important to note that the proxy only have access to the the access policy and cannot access the data because it doesn't have the user's private key. Next, the transformed ciphertext is uploaded to the Cloud.

When a user request to have access to the data, the Cloud starts by selecting an initial Fog node that manages one or more attributes of the data's access tree. The selected Fog node then tries to decrypt all the managed attributes with its transformed key (TK) (provided by the Trusted Authority). In the case the attribute is a collaboration attribute, the Fog checks for users (from the same group as the requesting user) to decrypt the attribute. If such a user is found, the collaboration attribute is decrypted. After that, the new access tree with the decrypted attributes along with the original one are sent to next selected Fog node. This process is repeated until all attributes are decrypted. Finally, the result is sent to the end-user which uses his private key (SK) to recover the original message. Further details are given in section 5.

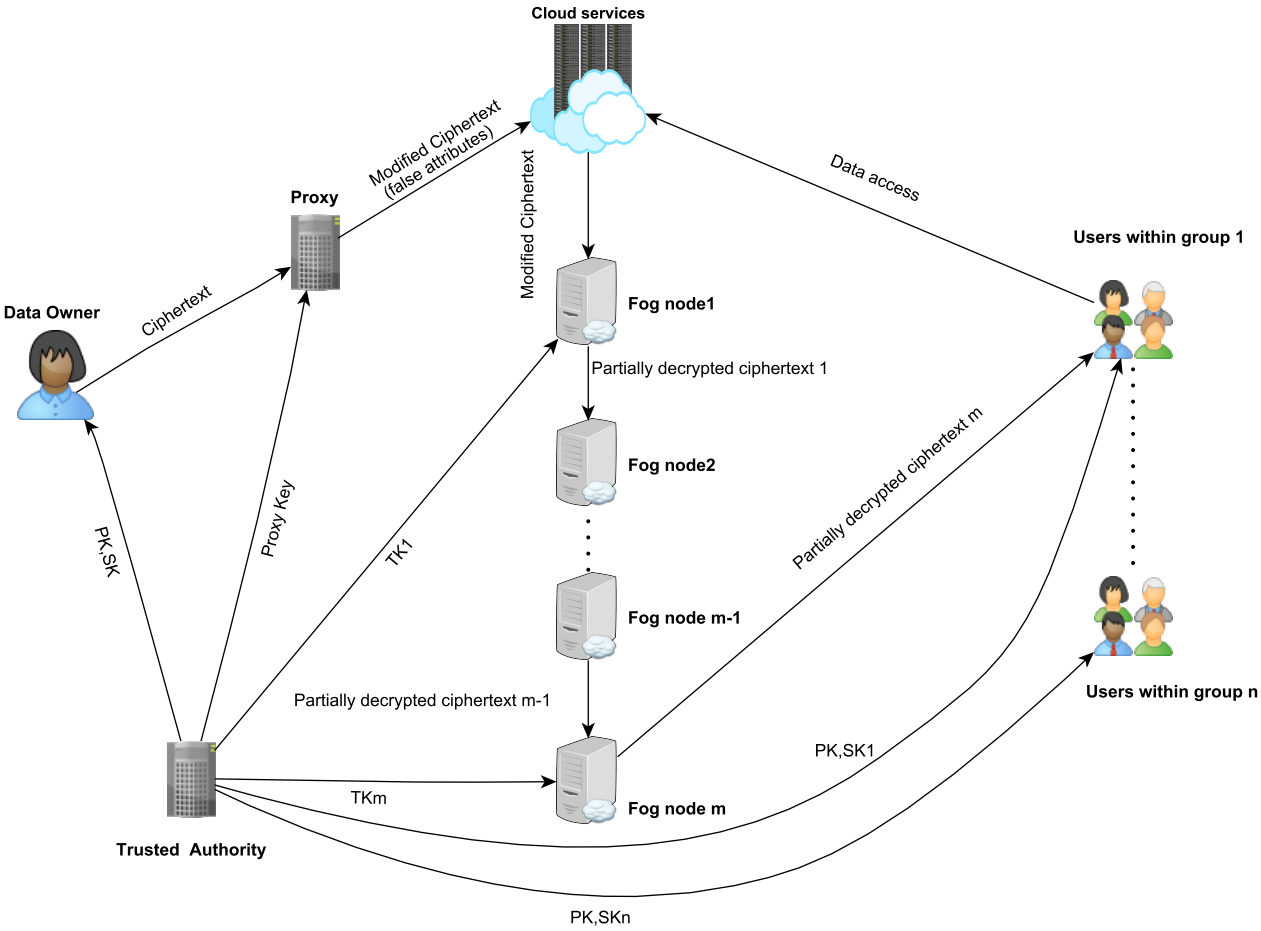

Figure 5.3: Scheme of the proposed solution[9]

5.3.1.2 Security assumption and Requirements

In our scheme, the security assumptions for the six roles can be defined as follow:

- The Cloud server, proxy, and Fogs are semi-trusted entities, They perform the tasks assigned to them, but try to get as much as possible information on both the outsourced data and the data owner.
- The trusted authority is assumed to be trusted (like in all systems using ABE).
- The data owner has access control to his outsourced data which are encrypted with

CP-ABE, and does not try to compromise his own confidential data.

The security model of our scheme covers the following aspects:

1. **Data confidentiality:** The content of the data must be confidential. Users not defined as recipients by the data owner, should not be able to access data, including Cloud, Fogs, the proxy, and users of the same group that collaborate by sharing their attributes.

2. **User collision resistance:** The collaboration is authorized in the same group only for collaboration attributes. Users cannot combine their non-collaboration attributes to satisfy the access policy,

3. **Collaboration outside of the group:** Attributes collaboration is authorized only inside the same group. Users outside the group cannot decrypt the access policy.

4. **Fine-grained access control:** The encrypted data can only be accessed by users whose attributes satisfy the access policy defined by the data owner. Such access policy must be flexible and expressive.

5. **Hidden access policy:** In the access policy, the attributes should be hidden so that the users cannot deduce sensitive information.

5.3.2 Different Phases Description

This work aims to allow the users in the same group to collaborate to get access to the data without compromising the access policy confidentiality and decrypt data with resource-constrained devices by delegating heavy computations from IoT to Fog.

5.3.2.1 Collaboration

We define two main rules for collaboration: (1) The users must be in the same group. If a user wants to collaborate with a user from a different group, access to data must

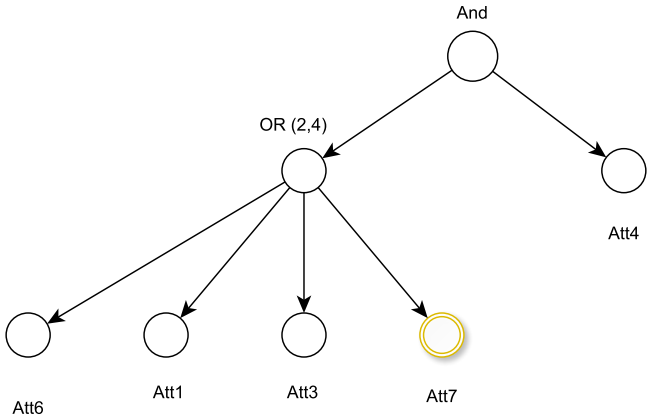

Figure 5.4: An Example of an Access Policy with a Collaboration attribute[9]

be denied. (2) the collaboration is performed only with what we call "collaboration attributes". These latter are defined by the data owner, which is included in the access policy.

We illustrate the proposed collaboration approach through an example. The data owner encrypts data with the access policy as in Figure 5.4. Two users in the same group want access to the encrypted file Figure 5.5: (1) the first user set of attributes is $S1$= {**Att1, Att4, Att6**}. This user can decrypt the ciphertext with his attributes directly because his attributes satisfy the access policy; (2) The second user has the set attributes defined by $S2$= {**Att3, Att4**}. In this case, this user cannot access the data, but needs the cooperation of other users (from the same group) that holds the rest of the needed attributes i.e: {**Att7**}.

Our idea to introduce collaboration attributes was inspired by [66], where the authors use transition nodes to perform collaboration. In our approach, we define our "collaboration node" which can be a leaf or non-leaf node. Users from the same group can collaborate only by using this type of node. To restrict the collaboration to users of the same group, we introduce a random secret key for each group, that will be added to the secret key of each user of that group.

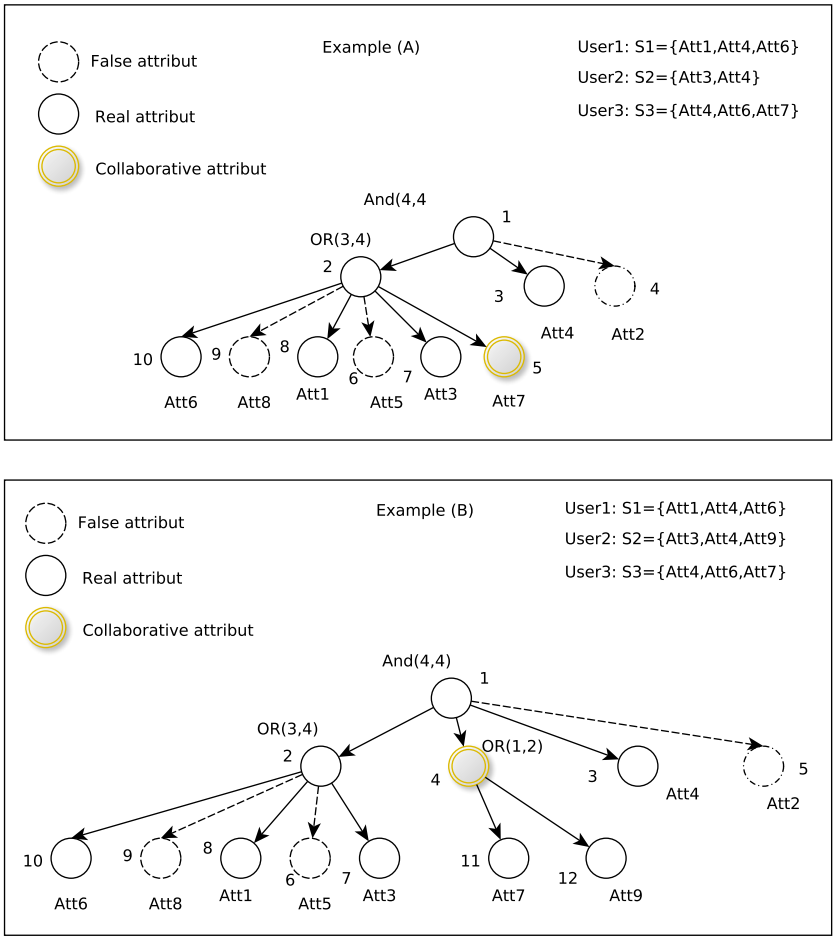

Figure 5.5: Examples of Access Policies with False Attributes[9]

5.3.2.2 Access Policy

Access structure: In the construction of our scheme, we use the standard access trees (example in Figure 5.4) which are used in the CP-ABE algorithm to describe the access policy [35]. This tree is constructed of several nodes, where each non-leaf node of the tree is a threshold gate that can be defined by these children and a threshold value. If num_x is the number of children of the $node_x$ and K_x is the threshold value, then $1 \leq K_x \leq num_x$. When $K_x = 1$, the threshold gate is an "OR" gate, and when $K_x = num_x$, it is an "AND" gate. We define the parent of a $node_x$ in the tree by the $parent(x)$. We define $att(x)$ if

the $node_x$ is a leaf node which is associated with an attribute. In addition, we define an $index(x)$ as the order of $node_x$ among these brothers where the T strategy tree defines an order among children of each non-leaf node, these children are numerated arbitrarily from 1 to num_x.

The concept is extended here to include collaborative nodes for controlled cooperation. A new function $R(x)$ is used to retrieve the users that can collaborate. For this, we assign each user a unique identifier; for example, the user i will have the identifier u_i. If there is a collaboration node, then $R(x)$ is executed as follows: If x is a leaf node, then $R(x)$ returns the attribute, the user identifier that holds this attribute and its path in the access tree. The path of node x of $att(x)$ can be obtained by defining indexes from the top of the tree to the leaf nodes. In this way, each attribute in the tree will have its path associated with the tree T. Consequently, an attribute must appear only once in the tree; otherwise, a new path must be generated for this attribute. For example, in figure 5.5, the attribute $attr$=**{Att7}** have the following path: $path_{attr}$=**{1,2,5}**. In the same figure, x is a non-leaf node, $R(x)$ returns a set of attributes with users identifier and their paths in the access tree T.

Access Structure with False Attributes: In CP-ABE, the access policy is sent in clear text with the ciphertext. A malicious user can obtain sensitive information from the access policy (like social security number, name, etc.). To resolve this issue, we add false attributes to the access policy as depicted in Figure 5.5. The false attributes are added by a proxy to reduce the processing at the data owner's level without compromising confidentiality. As show in the results, users do not know which attributes are real or false.

Table 5.1: Algorithm1 variable/function Description

Variable/Function	Description
AT	The original Access policy in tree form
current_Fog	The current Fog node used to decrypt AT
Query_bloom_filter(AT)	Selects a Fog that manages (not decrypted) attributes present in the Access policy
decrypted_AT	The partial decrypted of AT
attr	Attribute in the access tree
Get_attr (Fog)	Returns a set of attributes managed by the Fog node
attr.type	The type of the attribute "attr": can be either a collaboration or a standard attribute
collaboration_user_id	Contains the id of user who have the collaboration attribute
R(attr)	Finds the id of the user who have the collaboration attribute
decrypted_attr	Contains the value of decrypted attribute
Send_Receive(collaboration_user_id, attr)	Sends the collaboration attribute "attr" to the user "collaboration_user_id" for decryption then receives the result
Decrypt (current_Fog, attr)	Decrypts the attribute managed by "current Fog", returns $\perp$ if the attributes cannot be decrypted
Decrypt_Root_Nodes(decrypted_AT)	Decrypts root nodes if the condition on the leaf nodes is satisfied
Send(user, decrypted_AT)	Sends the final partial decrypted AT to the "user"
next_Fog	the next Fog node that will be used to decrypt the AT

Algorithm 1 Chained Collaboration Outsourcing Decryption Algorithm

```
1: Input: AT: Access tree;
2: current_Fog ← Query_bloom_filter(AT)
3: decrypted_AT ← ∅
4: while AT != ∅ do
5:     for attr ∈ Get_attr (current_Fog) do
6:         if attr ∈ AT then
7:             if attr.type = "Collaboration" then
8:                 collaboration_user_id ← R(attr)
9:                 if collaboration_user_id = Null then
10:                    Error()
11:                else
12:                    decrypted_attr ← Send_Receive(collaboration_user_id, attr)
13:                    if decrypted_attr = Null then
14:                        goto 8
15:                    decrypted_AT ← decrypted_AT ∪ {decrypted_attr}
16:                    AT ← AT − {attr}
17:                    if AT = ∅ then
18:                        Send(user, decrypted_AT)
19:                        Exit()
20:            else
21:                decrypted_attr ← Decrypt (current_Fog, attr)
22:                decrypted_AT ← decrypted_AT ∪ {decrypted_attr }
23:                AT ← AT − {attr}
24:                Decrypt_Root_Nodes(decrypted_AT)
25:                if AT = ∅ then
26:                    Send(user, decrypted_AT)
27:                    Exit()
28:    next_Fog ← Query_bloom_filter(AT)
29:    if AT != ∅ And next_Fog = Null then
30:        Error()
31:    else
32:        Current_Fog ← Next_Fog
33:    Send(next_Fog, AT, decrypted_AT)
```

5.3.2.3 Outsourcing

Knowing that the decryption cost in the CP-ABE algorithm increases with the complexity of the access policy, we propose to outsource the decryption process to Fog nodes. These latter collaborate to help decrypting the data. In addition to the Fog nodes, users which have the collaboration attributes also participate in the decryption process.

The decryption is performed with an intermediate key TK_i created by TA for each Fog i. To limit the communication between the end-user and the Fogs, we propose a new "chain scheme". In the latter, each time a Fog node finishes the partial decryption, the result is sent to another Fog node that will continue the partial decryption. We use the Bloom filter [72] method which was proposed by Burton Howard used to check whether given data exist in a data set. In the latter, each Fog manages a set of attributes. After the partial decryption process is finished and if there are non decrypted attributes, the current Fog node queries the bloom filter to search for Fog nodes that can satisfy any of the remaining attributes. The proposed "Chained Collaboration Outsourcing" process can be illustrated by algorithm 1. The Cloud starts by selecting an initial Fog node that has one or more attributes contained in the access tree (by querying the bloom filter) (line 2). Then, the current Fog node tries to decrypt all the attributes that it manages (lines 5 to 23). If the attribute is a collaboration attribute, the Fog checks for users that can decrypt the attribute (lines 7 to 14). If it finds such a user, it collaborates with him to decrypt the attribute using the R function defined in section 5.3.2.1. Otherwise, it decrypts the attribute locally (lines 21 to 24). After decrypting all the attribute managed by the current Fog (lines 25 to 27), it selects a new Fog node (line 28) and sends the new access tree with the decrypted attributes along with the original one (line 33). An error is returned if there are still attributes that no Fog node or collaboration user can decrypt (lines 9,10 and 29, 30). This process is repeated until all attributes are decrypted. Finally, the result is sent to the end-user (lines 25, 26).

Approach Details:

In this part, we describe -in detail- eight phases of our model.

- $Setup \rightarrow (PK, MSK)$ **:** This algorithm is executed by the TA to generate a Public key (PK) and a Master key (MK). The algorithm takes no input and chooses a bilinear group G_1 of prime order p with a generator g. The universal attribute set N, is a string labeled from 1 to i, where i is the number used to index the attributes. The authority chooses uniformly at random G_1, $h_1 \ldots h_i$. In addition, it chooses an exponent random $\alpha, a, \beta_2 \in Z_p$. Finally, the public key is obtained by:

  $PK = \{e(g,g)^{\alpha}, g, g^a, \{h_i\}_i, g^{\beta_2}\}$

  And the master key by:

  $MSK = \alpha, \beta_2, a$

  In our work, we assume that each user is assigned to a group. For each group, the TA generates a unique random variable $\theta \in Z_p$. This key indicates to which group does a specific user belongs. Moreover, we suppose there are L groups.

- $KeyGen(PK, MSK, W) \rightarrow (SK)$**:** According to the attribute set w submitted by the user, the TA generates the secret key corresponding to the user as follow: First, the TA verifies the user's group. Then, it retrieves the private secret of the group θ after that the TA chooses $t \in Z_p$ randomly. Finally, the TA creates the private key as follow :

  $SK = \{K = g^{(\alpha + at)\theta}, L = g^{t\theta}, \{k_j = h_j^t\} \forall j \in w, E_i = g^{(t\theta/\beta_2)}\}$

  Where E_i is the translation key used in collaboration.

- $Transform(SK) \rightarrow (TK, SK')$**:** When the user wants to access data, the TA creates the transformation key for the Fogs, which will later help the user partially decrypt the data. For this, the TA chooses a random element $\delta \in Z_p$ and returns the following transformation key:

$$TK = \{K = g^{(\alpha+at)\theta\delta}, L = g^{t\theta}, \{k_j = h_j^t\} \forall j \in w, E_i = g^{(t\theta/\beta_2)}\}$$

It also generates the intermediate secret key of the user defined by: $SK' = \delta$.

- $Encryption(M, PK, T) \rightarrow (CT)$**:** Let M be a message that the data owner wants to share. First, the data owner encrypts the message using a symmetric algorithm like AES. After that, the data owner chooses a symmetric key as a random variable $\mu \in Z_p$. The encrypted message is denoted as $E\mu(M)$.

Finally, the data owner encrypts the symmetric key μ by using the access policy. The data owner defines an access tree T to formulate the access policy. He also defines the collaboration nodes in the access policy on the basis that he/she can authorize several users to collaborate to satisfy the access policy.

First, the algorithm chooses a polynomial $q(x)$ for each node x in the policy tree T. These polynomials are chosen from top to down, starting from the root node r. For each non-leaf node x in the tree, the degree $d(x)$ is defined by $d(x) = k(x) - 1$. Then, starting with the root node of the policy tree, the algorithm randomly chooses $s \in Z(p)$ and sets $q_r(0) = s$. After that, it randomly chooses $d(r)$ other points of the polynomial q_r to define it entirely. Subsequently, it sets for other node x(including both leaf nodes and non-leaf nodes) $q_x(0) = q_{parent}(x)(index(x))$ and randomly chooses d_x other points and completely defines q_x. The degree of leaf nodes is set to be 0. Once all the polynomials have been defined we put $\lambda_x = q_x(0)$ for each x in T. Also, we choose a set of n random number $\{r_1 \dots r_n\}$ where each $r_i \in Z_p$. For

each λ_i, r_i the data owner computes $C_{(1,i)} = g^{\lambda_i a}.h_{\pi(i)}^{-r_i}, C_{(2,i)} = g^{r_i}$. We note that g^a and $h_{\pi(i)}$ are present in the public key. Finally, the ciphertext is defined by:

$$CT = \{T, C_0 = Ke(g,g)^{s\alpha}, c^{'} = g^s, \forall y \in Y \ \{C_{(1,y)}, C_{(2,y)}\}, \forall x \in X \ \{C_{(4,x)} = g^{\beta_2 \lambda_i a}\}\}$$

Where Y denotes the set of a real (i.e: not false) leaf node in T. And X denotes the set of collaborative leaf nodes in T.

- $CT\_transform(CT) \rightarrow (CT')$**:** After creating the access policy, the data owner sends CT to the proxy. The proxy chooses a random value Z and shares it using the same secret sharing scheme algorithm that is used to share the secret s. We obtain $z_x = q_x(0)$ for each x in T. After that, the proxy creates $key = g^z$ for each x in T. key is used so that the users cannot deduce the real attributes from the Access policy.

  The proxy calculates:

  $\{C_{(1,i)}.g^{z_i}, C_{(2,i)}.g^{z_i}\}$ if the attributes $\in T$

  And

  $\{C_{(3,i)} = g^{z_i} h_{\pi(i)}^{-r_i}\}$ if the attributes $\notin T$

  Finally:

$CT' = \{T', C_0 = Ke(g,g)^{s\alpha}, C' = g^s, C" = C' g^{z_i}, \{C_{(1,i)} g^{z_i}, C_{(2,i)} g^{z_i}\}$ if $i \in T, \{C_{(3,i)} = g^{z_i} h^{-r_i}\}$ if $i \notin T, C_{(4,i)} = g^{\beta_2 \lambda_i a}\}$

Where $C_{(4,i)}$ is the collaboration node.

- $Fog\_Decrypt(CT', PK, TK) \rightarrow (CT")$**:** When the Fog, e.g., Fog_i receives CT', it tries to derypt the maximum number of attributes it manages using TK. If the attribute (x) is not supported by the Fog, then the attribute will be outsourced to another Fog. In this case, Fog_i queries the bloom filter to find another Fog that can satisfy the attribute. If the attribute (x) is supported by Fog_i then Fog_i call $DecryptNode$, The Fog executes $DecryptNode$ as follow : If the x node $\in T$ then the Fog_i execute :

$DecryptNode(x) = e(C_{(1,i)}, L) e(C_{(2,i)}, K) = e(g^{z_i} g^{\lambda_i a} h^{-r}, g^{t\theta}) e(g^{r_i}, h_i^t) = e(g^{z_i} g^{\lambda_i a} h^{-r}, g^{t\theta}) e(g^{t\theta}, h^{t_i}) = e(g^{z_i} g^{\lambda_i a}, g^{t\theta})$
$= e(g^{z_i}, g^{t\theta}) e(g^{\lambda_i a}, g^{t\theta})$

If the attribute is a false node, the Fog executes $DecryptNode$ as follow:

$DecryptNode(x) = e(C_{(3,i)}, L) e(C_{(2,i)}, K_{\pi(i)}) = e(g^{z_i} h^{-r_i}, g^{t\theta}) e(g^{r_i}, h_i^t) = e(g^{z_i} h_i^{-r}, g^{t\theta}) e(g^{t\theta}, h^{r_i}) = e(g^{z_i}, g^{t\theta}) = e(g,g)^{z_i t\theta}$

If the node is a collaboration node, then the collaborate user executes the $DecryptNode$, then it makes the transition of the node as follows:

$R_z = (C_{(4,i)}C_{(5,i)}, E_i/E_i^{'}).R_z^{'}$

$= (g^{\beta_2\lambda_i a}g^{\beta_2 z_i}, g^{(t\theta - t^{'}\theta)/\beta_2})e(g,g)^{t'\theta\lambda_i a}.e(g,g)^{z_i t'\theta}$

$= (g^{\beta_2\lambda_i a}, g^{(t\theta - t^{'}\theta)/\beta_2}.e(g,g)^{t^{'}\theta\lambda_i a}.e(g^{\beta_2 z_i}, g^{(t\theta - t^{'}\theta)/\beta_2}).e(g,g)^{z_i t'\theta}$

$= e(g,g)^{t\theta\lambda_i a}.e(g,g)^{z_i t\theta}$

After that, the Fog computes the non-leaf nodes in a Bottom-up manner.

For a non-leaf node(x), the Fog computes the x value as follow :

$$V_x = \prod_{x\in Q}[e(C_{(1,i)}, L)e(C_{(2,i)}, K)]^{C_i} \cdot \prod_{x\in Q'}[e(C_{(3,i)}, L)e(C_{(2,i)}, K))]^{C_i} \cdot \prod_{x\in Q"} R_z^{C_i}$$

$$= \underbrace{\prod_{x\in Q}[e(g^{z_i}, g^{t\theta}).e(g^{\lambda_i a}, g^{t\theta})]^{C_i}}_{1} \cdot \underbrace{\prod_{x\in Q'}[e(g,g)^{z_i t\theta}]^{C_i}}_{2} \underbrace{\prod_{x\in Q"}[e(g,g)^{t\theta\lambda_i a}.e(g,g)^{z_i t\lambda}]^{C_i}}_{3}$$

We calculate the first part of the equation :

$(1) = \prod_{x\in Q}[e(g,g)^{t\theta z_i}.e(g,g)^{t\theta\lambda_i a}]^{C_i} = e(g,g)^{t\theta\sum z_i C_i}.e(g,g)^{t\theta a\sum\lambda_i' C_i'}$

Therefore, we can compute a set of $C_i's \in Z_p$ by applying Gauss-Jordan elimination such that $\sum\lambda_i' C_i' = s'$ and $\sum z_i' C_i' = Z'$

The second part is calculated as follow:

$(2) = e(g,g)^{t\theta p'}.e(g,g)^{t\theta s'}$

$= \prod_{x\in Q'}[e(g,g)^{z_i" t\theta}]^{C_i"} = e(g,g)^{t\theta\sum z_i" C_i"} = e(g,g)^{t\theta z_i"}$

Knowing that: $\sum z_i"C_i" = Z"$

Finally the calculation of the third part is

$(3) = \prod_{x \in Q"} [e(g,g)^{t\theta\lambda_i'''a}.e(g,g)^{z_i'''t\theta}]^{C_i'''} = e(g,g)^{at\theta s"}e(g,g)^{t\theta z_i'''}$

Knowing that: $\sum \lambda_i'''C_i'' = S"$ and $\sum z_i'''C_i'''' = Z'''$ where $S' + S" = S$ is the sharing and $Z + Z + Z''' = Z$ is the sharing secret Z.

Finally, the Fog computes :

$CT" = e(g,g)^{t\theta Z'}e(g,g)^{t\theta s'}.e(g,g)^{t\theta Z"}e(g,g)^{t\theta s"}e(g,g)^{t\theta Z'''} = e(g,g)^{t\theta Z}e(g,g)^{at\theta s}$

Otherwise, the value of x is set to $\perp$.

Fog_i transmits the intermediate result to the next Fog until no Fog can satisfy the access policy. If the latter is satisfied then the decryption process is stopped and last Fog transmits the decrypted result to the end-user to recover the message with his private key.

- $Cloud\_Decrypt(PK, CT', TK)$: In the same time, the Cloud computes and send the result to the user :

$$
\begin{aligned}
& e(C', K) \prod_{x \in Q"} [(e(C", L))/(e(C', L))]^{C_i} \\
&= e(g^s, g^{(\alpha+at)\theta\delta}) \prod_{x \in Q"} [e(g^s g^{z_i}, g^{t\theta})/e(g^s, g^{t\theta})]^{C_i} \\
&= e(g,g)^{\theta\delta s(\alpha+at)} \prod_{x \in Q"} [e(g^s, g^{t\theta})e(g^{z_i}, g^{t\theta})/e(g^s, g^{t\theta})]^{C_i}
\end{aligned}
$$

$= e(g,g)^{\theta\delta s\alpha}.e(g,g)^{\delta t\theta as}.\prod_{x\in Q"} e(g^{z_i}, g^{t\theta})^{C_i}$

$= e(g,g)^{\theta\delta s\alpha}.e(g,g)^{at\theta s\delta}.e(g,g)^{t\theta\sum z_i C_i}$

$D = e(g,g)^{\theta\delta s\alpha}e(g,g)^{at\theta s\delta}e(g,g)^{t\theta Z}$

- $User\_Decrypt(D, CT", SK', \theta) \rightarrow (M)$**:** when the user receives partial decryption from Fog and Cloud, he computes:

$$\frac{Me(g,g)^{\alpha s}}{\frac{e(g,g)^{\theta\delta s\alpha}.e(g,g)^{at\theta s\delta}.e(g,g)^{t\theta Z}}{(e(g,g)^{t\theta Z}.e(g,g)^{at\theta s}}}$$

$$= \frac{Me(g,g)^{\alpha s}}{\left(\frac{e(g,g)^{\theta\delta s\alpha}.e(g,g)^{ats\theta\delta}}{e(g,g)^{at\theta s}}\right)^{1/\theta}}$$

$$= \frac{Me(g,g)^{\alpha s}}{\frac{(e(g,g)^{\delta s\alpha}.e(g,g)^{ats\delta})^{1/\delta}}{e(g,g)^{ats}}}$$

$$= \frac{Me(g,g)^{\alpha s}}{\frac{e(g,g)^{s\alpha}.e(g,g)^{ats}}{e(g,g)^{ats}}}$$

$$= \frac{Me(g,g)^{\alpha s}}{e(g,g)^{\alpha s}} = M$$

Rq: the decryption process in the constrained device is reduced to one exponentiation operation

5.4 Security Analysis

In this section, we discuss the security properties of the proposed approach.

- **Data privacy:** Confidentiality requires that the Cloud, Fog, and the unauthorized users (including the collaborating users) cannot decrypt the data. In the encryption

process, the proxy adds false attributes to the access policy, but it cannot access the data without having the user's private key. In other words, these entities must remove $e(g,g)^{\alpha s}$ from the ciphertext to be able to calculate the paring. However, this paring value is blinded by the private secret key of the user (i.e: δ). As a result, unauthorized parties cannot recover the message even if they have access to $e(g,g)^{\alpha s}$.

- **Collision resistance:** User resistance to collision means that the users cannot collude their key to access data individually. In our solution, the collaboration is allowed only on the collaboration node (which is defined by the data owner) of the Access tree and in the same group. In SHARE-ABE, the TA generates different keys for each user. Each key is associated to a random value t, which is unique for each user and makes the combination of the different keys insignificant. In addition, the TA generates for each group a random key, which makes the collaboration of users in different groups impossible. This makes the proposed scheme resistant to collision.

- **Hidden access control:** In traditional ABE, the access policy is sent clear along with the ciphertext, making it vulnerable to information theft. In SHARE-ABE, the proxy adds false attributes to the access policy to obfuscate the real attributes. With this method, a malicious user cannot guess which are the real attributes. Later, the false attribute is eliminated in the decryption process. Additionally, the entities that participate in the decryption process (like the Fog and the Cloud) cannot know which attribute was used in the decryption process (because of the introduced false attributes). Moreover, in our chain decryption scheme, each Fog does not know the attributes of the previous Fog, because the attributes which were decrypted by the precedent Fog are replaced by their value, which looks just like any random value.

5.5 Performance Evaluation

In this section, we present the analysis of performances of our proposed scheme . We start by giving the experiments setting . Then, we compare our solution to the related works in terms of overhead, and execution time.

5.5.1 Computation Cost

We use the same notations and metrics as the previous approach (i.e section 4.4). then we compare our work and the different approaches both at the End-user and Fog levels.

5.5.1.1 At the End-user Level

End-user computations are depicted in Table 5.2. In chapter 4, section 4.4.1.1 we showed that our first contribution [47] outperfoms tradictional ABE apprachhes. compared to Traditional CP-ABE, [66] [62] and [47] that have complexity of $(2n+1)p+(n+nl)E$, $(2ny+nc+2)p+(ny+nl)E$, $3P+2(n)E$, $(1E)$ respectively. SHARE ABE has a complexity of $(2E)$ which is better than the reviewed approaches where these latter do not support group collaboration, whereas ours does.

Table 5.2: Computation Costs at the End-user Level

Scheme	Complexity of the Decryption Process
CP-ABE	$(2n+1)p+(n+nl)E$
Wang et al [62]	$3P+2(n)E$
Belguith et al [61]	$(1E)$
Xue et al [66]	$(2ny+nc+2)p+(ny+nl)E$
Zhao et al [64]	$(2E)$
Tu et al [69]	$(1E)$
Collaborative approach	$(1E)$
SHARE-ABE	$(2E)$

5.5.1.2 At the Fog-nodes Level

In the Share ABE scheme, the Fog nodes are charged with expensive computation operations. Also, the computation on the Fog node level is reduced by introducing the collaboration between Fogs , where the set of attributes is assigned to each Fog. Table 5.3 shows the decryption computation cost of the proposed approaches at the Fog node level (which is given by $(2nr_i + 2nf_i + nc_i)p + 3|Ln_i|E)$ is better than [64] and [69]. Table 5.4 also shows that compared to our collaboration approach(Fog in the parallel scheme) that generates many interactions between Fogs and end-users (which is given by 2x(Nf + 1)), which results in a growing expense of networking and energy use, our approach optimizes networking and energy by minimizing the number of communications needed to be performed in order for the end-user to decrypt the ciphertext.

Table 5.3: Computation Costs at the third party server Level

Scheme	Complexity of the Decryption Process
Tu et al [69]	$(2 + 2n)P + E$
Zhao et al [64]	$(k\|\alpha\| + \|\alpha\|)P + (k\|\alpha\| + \|\alpha\|)E + P + E$
Collaborative approach (Fog)	$(2nr_i + 2nf_i + nc_i)p + 3\|Ln_i\|E$
SHARE-ABE (Fog)	$(2nr_i + 2nf_i + nc_i)p + 3\|Ln_i\|E$

Table 5.4: Number of Communications between the End-user and Entities in the Decryption Process

Scheme	Number of Communications
Collaborative approach	$(2 \times (N_f + 1))$
SHARE-ABE	$(2 + N_f)$

5.5.2 Evaluation

We used Charm [70] to implement our schemes and the related approaches. Charm is a framework based on phyton that facilitates the rapid prototyping of cryptographic

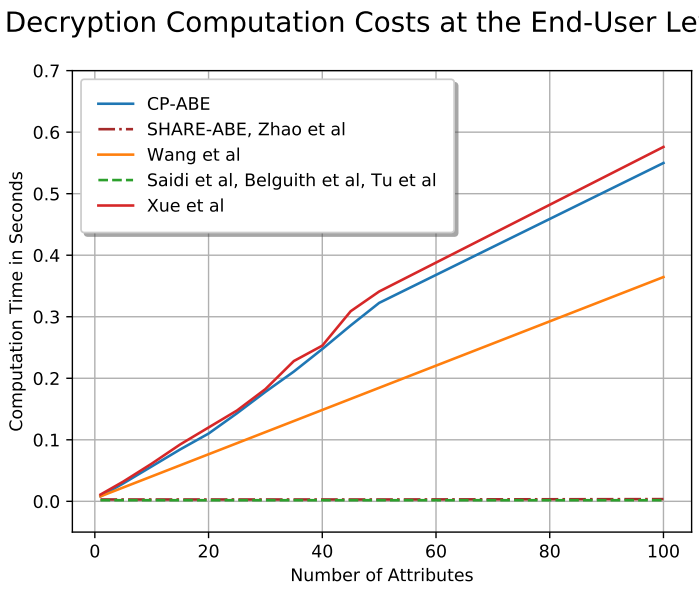

Figure 5.6: Comparison of Decryption Computation costs at the End-User Level.

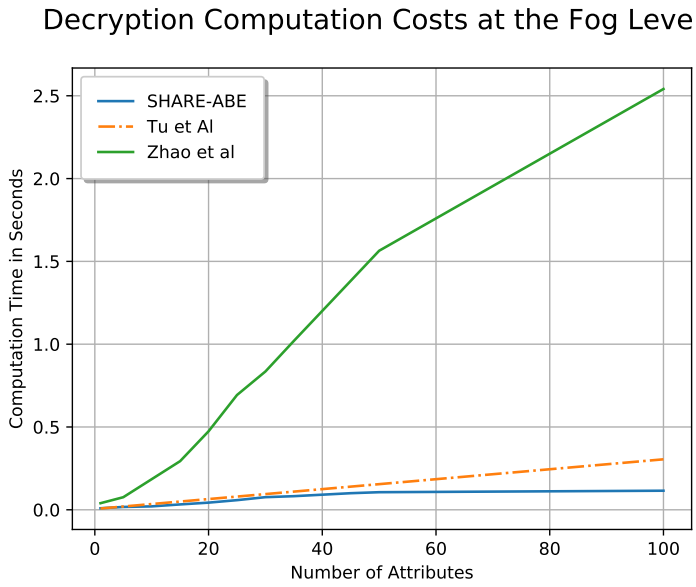

Figure 5.7: Decryption Computation Costs at the Fog Level.

schemes and protocols. The experiments are conducted on a machine with a Xeon E5-2630 (2,20GHz) processor and 32 Go RAM. We used a Virtual Machine with one core and 2.00 GB of RAM. The VM runs the Ubuntu16.04 64-bit OS with python 2.7. The number of attributes used in the experiments are L ={ 1, 5,10, 15, 20, 25, 30, 35, 40, 45, 50, 100}. In the experiments, we evaluate the time of one multiplication, one pairing, and one exponentiation. The time of the multiplication operation is very short. Thus it can be ignored in the analysis. The time of a single pairing operation was evaluated to 1.5 ms, and the time of one exponentiation operation is on average 1.8 ms. In our experiments, we evaluate the time cost of data decryption and compare it with the related approaches, including traditional CP-ABE, Wang et al.[62], Belguith et al.[61], Xue et al.[66], Saidi et al.[47], Zhao et al.[64] and Tu et al.[54].

Figure 5.6 shows the decryption computation time on the user's side with respect to the number of attributes in the access policy. The figure shows that in CP-ABE, Wang et al.[62] and Xue et al.[66], the decryption time on data owners grows as the number of attributes in access policy increases, as opposed to Belguith et al.[61], Saidi et al.[47], Tu et al.[54], Zaho et al [64] and SHARE-ABE, where the decryption times do not depend on the number of attributes, this is because the decryption operations are outsourced. We can see that SHARE-ABE and Zhao et al.[64] show similar results. However, the approach of Zhao et al does not support collaboration and group data sharing. Finally, the decryption times in Belguith et al.[61], Saidi et al.[47] and Tu et al.[54] are slightly lower than SHARE-ABE. This is mainly because in SHARE-ABE, we first test if the user belongs to the group before performing the actual decryption.

We also evaluate the decryption time on the Fog side. The results are shown in Figure 5.7. In the experiments, we use three Fog nodes for SHARE-ABE. We compare our work to approaches that use Fog outsourcing, i.e., Tu et al.[54] and Zhao et al.[64]. We observe that the experimental results in these approaches, including the proposed SHARE-ABE follow a linear relationship (approximately) as the number of attributes

increases. In Figure 5.7, it is clear that our approach gives the best results. This is because we divide the set of attributes over several Fogs, where each Fog node manages a specific set of attributes.

5.6 Conclusion

In this chapter, we presented SHARE-ABE, a novel collaborative approach based on CP-ABE. The approach preserves privacy and uses Fog computing to outsource decryption operations. By enforcing a new chained collaboration method between multiple Fogs, our solution reduces computational decryption and networking cost and is well adapted for resource-constrained devices environments. Indeed, in this chained collaboration, each Fog decrypts the access policy based on the attributes it manages and passes the results to the next until the user receives the decrypted message. On the other hand, in our scheme, the data owner can allow users -through the access policy- to collaborate on certain attributes to satisfy the access policy. This collaboration is permitted only for users in the same group. To restrict the collaboration to users of the same group, we introduced a random secret key for each group. This latter is added to each user's private key. A new function is introduced to retrieve users who can collaborate in the same group when collaborative nodes are used in the access policy. Furthermore, our solution used a proxy to incorporates false attributes to preserve the privacy of the access policy. The experiments also demonstrate that the proposed approach outperforms state-of-the-art approaches and gives encouraging results.

Chapter 6

Conclusion and Perspectives

6.1 Summary

Nowadays, there is an exponential growth in connected and heterogeneous devices. As a result, there is a need for creating an architecture capable of handling, analyzing, and storing the huge amount of data generated by these devices. Cloud computing already satisfies the bulk of the requirements for addressing this colossal technological growth. Cloud-based solutions, on the other hand, have significant limits in terms of real-time processing, fast data response, and latency issues. Consequently, a novel architecture known as Fog computing was created to bring Cloud capabilities closer to the network's edge. Despite the advantages of both systems, many hurdles must be are yet to be addressed before using either. The purpose of this thesis is to examine data security issues especially those associated with externalization technologies such as Cloud and Fog computing. First, we have briefly reviewed the architectures of Cloud and Fog computing and the security mechanisms used to implement security services such as confidentiality, authentication, privacy, etc. One of the most robust countermeasures that may be used to protect data-sharing processes in Cloud computing is cryptographic access control implementing attribute-based encryption(ABE). However, this method has some shortcomings. One of the most important one is that encryption and decryption are resource-consuming. This is a serious issue when devices have limited of CPU power, memory, and energy. In

addition, in traditional ABE, the access policy is sent in the clear text along with the Cipher-text. Another limitation is that it does not consider user collaboration, as they only allow assigning one access authorization to one user. I this work, we started by identifying the various solutions proposed in the literature concerning attribute-based access control paradigm related to (1) the outsourcing the encryption /decryption process, (2) the protection of the access policy, and (3) the collaboration among users. Then, using CP-ABE, we presented a novel method for offloading computation overhead from resource-constrained devices to Fog nodes. Our method leverages the collaboration of Fog nodes (in parallel architecture) and a newly proposed partial decryption technique with a concealed access policy to achieve reduced computation overhead while maintaining safe and fine access control. We demonstrated that our approach is significantly more efficient than the original CP-ABE through simulations. Additionally, we have suggested SHARE-ABE, an original collaborative encryption based on group-oriented data sharing. Fog nodes collaborate to partially decrypt the data using an original and efficient chained architecture. Furthermore, we introduce a new construction of a collaboration attribute that allows users within the same group to combine their attributes while satisfying the access policy. Experiments and analyses of the security properties demonstrate that the proposed scheme is secure and efficient, especially for resource-constrained devices..

6.2 Perspectives

We identified four main directions for our future works. First, we intend to apply the same group collaboration solution to our first proposition, which aims to divide the tree into several sub-trees and compare the results with the SHARE-ABE approach. We also intend to address the outsourced encryption process by using a similar scheme to our proposed Fog computing architecture. However, CP-ABE is not only known to be complex in the decryption but also the encryption has high overhead. CP-ABE uses complex cryptographic operations to encrypt a message. Similarly to the proposed decryption

solution it can be interesting to develop practical solutions for outsourcing encryption to the third party to reduce the complexity of encryption at the user level. Besides, we also intend to address the bottleneck problem in our proposed scheme where a single authority manages all attributes. This problem appears when the data owners share their data under access policies described via attributes issued from many domains and organizations; the single authority may create a bottleneck performance issue to overcome the issue mentioned above, it may be necessary to integrate a Multi-Authority CP-ABE in our solution. In the Multi-Authority CP-ABE schemes, the users can possess attributes issued by various attribute authorities. Data owners may additionally encrypt their data using access rules established across multiple attribute authorities. Because attributes are maintained separately by various attribute authorities in MA-CP-ABE systems, both computing and trust are spread between several attribute authorities rather than on a single one.

Finally, further research and consideration should be given the revocation and management of a wide range of attributes. Indeed, When a user's performance deteriorates or if he leaves the system, the corresponding access control scheme must decrease or remove the user's access privileges without incurring the substantial computational cost. Changing attributes in ABE can be difficult to handle since modifying a single property may significantly affect a considerable number of users that access the same attribute.

List of Publications

1. Saidi, A., Nouali, O. and Amira, A. SHARE-ABE: an efficient and secure data sharing framework based on ciphertext-policy attribute-based encryption and Fog computing. Cluster Comput (2021). https://doi.org/10.1007/s10586-021-03382-5

2. Saidi, A., Nouali, O. and Amira, A. Collaborative and fast decryption using Fog computing and a hidden access policy. CS and IT Conference Proceedings. pages 57-71. 11 2019.

References

[1] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, pages 457–473, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. 3, 32, 34, 37

[2] SinghAshish and ChatterjeeKakali. Cloud security issues and challenges. *Journal of Network and Computer Applications*, 2017. 8

[3] Cearley D. Plummer, D. and D. Smith. Cloud computing confusion leads to opportunity. Technical report G00159034, Gartner Research, 2008. 9

[4] Al Bento and A. Aggarwal. Cloud computing service and deployment models: Layers and management. 2012. 9

[5] R. Buyya, J. Broberg, and A. Goscinski. Cloud computing principles and paradigms. 2011. 9

[6] P. Mell and T. Grance. Sp 800-145. the nist definition of cloud computing. 2011. 9

[7] D. Nurmi, R. Wolski, Chris Grzegorczyk, Graziano Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The eucalyptus open-source cloud-computing system. *2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 124–131, 2009. 10

[8] Michaela Iorga, Larry Feldman, Robert Barton, Michael Martin, Nedim Goren, and Charif Mahmoudi. Fog computing conceptual model, 2018-03-14 2018. 13

[9] Ahmed Al Saidi, Omar Nouali, and Abdelouahab Amira. Share-abe: an efficient and secure data sharing framework based on ciphertext-policy attribute-based encryption and fog computing. *Cluster Computing*, 2021. 14, 76, 82, 87, 89, 90

[10] S. Sarkar and S. Misra. Theoretical modelling of fog computing: a green computing paradigm to support iot applications. *IET Networks*, 5:23–29, 2016. 14

[11] Mina Deng, Milan Petkovic, Marco Nalin, and Ilaria Baroni. A home healthcare system in the cloud–addressing security and privacy challenges. In *2011 IEEE 4th International Conference on Cloud Computing*, pages 549–556, 2011. 16

[12] Ahmed Lounis, Abdelkrim Hadjidj, Abdelmadjid Bouabdallah, and Yacine Challal. Secure medical architecture on the cloud using wireless sensor networks for emergency management. In *2013 Eighth International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 248–252, 2013. 16

[13] Gregorio LÃşpez, VÃıctor Custodio, and JosÃľ Ignacio Moreno. Lobin: E-textile and wireless-sensor-network-based platform for healthcare monitoring in future hospital environments. *IEEE Transactions on Information Technology in Biomedicine*, 14(6):1446–1458, 2010. 16, 18

[14] Meng Ma, Yu Huang, Chao-Hsien Chu, and Ping Wang. User-driven cloud transportation system for smart driving. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 658–665, 2012. 16

[15] Swarun Kumar, Shyamnath Gollakota, and Dina Katabi. A cloud-assisted design for autonomous driving. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, page 41âĂŞ46, New York, NY, USA, 2012. Association for Computing Machinery. 16

[16] Swarun Kumar, Lixin Shi, Nabeel Ahmed, Stephanie Gil, Dina Katabi, and Daniela

Rus. Carspeak: A content-centric network for autonomous driving. *SIGCOMM Comput. Commun. Rev.*, 42(4):259âĂŞ270, August 2012. 16

[17] OpenFog Consortium Architecture Working Group et al. Openfog reference architecture for fog computing. *OPFRA001*, 20817:162, 2017. 18

[18] Razvan Craciunescu, Albena Mihovska, Mihail Mihaylov, Sofoklis Kyriazakos, Ramjee Prasad, and Simona Halunga. Implementation of fog computing for reliable e-health applications. In *2015 49th Asilomar Conference on Signals, Systems and Computers*, pages 459–463, 2015. 18

[19] Jitender Grover, Shikha, and Mohit Sharma. Cloud computing and its security issues : A review. In *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–5, 2014. 19

[20] Akshita Bhandari, Ashutosh Gupta, and Debasis Das. A framework for data security and storage in cloud computing. In *2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*, pages 1–7, 2016. 19

[21] Chunming Rong, Son Thanh Nguyen, and Martin Gilje Jaatun. Beyond lightning: A survey on security challenges in cloud computing. *Comput. Electr. Eng.*, 39:47–54, 2013. 20

[22] Chunming Rong and Son T. Nguyen. Cloud trends and security challenges. In *2011 Third International Workshop on Security and Communication Networks (IWSCN)*, pages 1–7, 2011. 22

[23] A. A. Alli and M. M. Alam. The fog cloud of things: A survey on concepts, architecture, standards, tools, and applications. *Internet Things*, 9:100177, 2020. 24

[24] Mark-Shane Scale. Cloud computing and collaboration. *Library Hi Tech News*, 26:10–13, 2009. 25

[25] Danan Thilakanathan, Shiping Chen, Surya Nepal, and Rafael Alejandro Calvo. Secure data sharing in the cloud. 2014. 25

[26] Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou. Achieving secure, scalable, and fine-grained data access control in cloud computing. *2010 Proceedings IEEE INFOCOM*, pages 1–9, 2010. 25

[27] Huang Qinlong, Ma Zhaofeng, Yang Yixian, Niu Xinxin, and Fu Jingyi. Improving security and efciency for encrypted data sharing in online social networks. *China Communications*, 11:104–117, 2014. 25

[28] Kyle Chard, Steven Tuecke, and Ian T. Foster. Efficient and secure transfer, synchronization, and sharing of big data. *IEEE Cloud Computing*, 1:46–55, 2014. 25

[29] Sana Belguith, Nesrine Kaaniche, and Mohammad Hammoudeh. Analysis of attribut based cryptographic techniques and their application to protect cloud services. *Transactions on Emerging Telecommunications Technologies*, 2019. 25, 26

[30] Asmaa Fowzi Alotaibi, Ahmed Barnawi, and M. I. Buhari. Attribute based secure data sharing with efficient revocation in fog computing. *Journal of Information Security*, 08:203–222, 2017. 26, 27

[31] R. Shirey. Internet security glossary, version 2. *RFC*, 4949:1–365, 2007. 29

[32] Vincent C. Hu, David F. Ferraiolo, R. Kuhn, Adam Schnitzer, Kenneth Sandlin, R. Miller, and K. Scarfone. Guide to attribute based access control (abac) definition and considerations. 2014. 31

[33] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, 2001. 32

[34] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS âĂŹ06, page 89âĂŞ98, New York, NY, USA, 2006. Association for Computing Machinery. 33, 39

[35] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. *2007 IEEE Symposium on Security and Privacy (SP '07)*, pages 321–334, 2007. 33, 38, 39, 40, 90

[36] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Kaitai Liang, Hui Ma, and Lifei Wei. Auditable σ -time outsourced attribute-based encryption for access control in cloud computing. *IEEE Transactions on Information Forensics and Security*, 13:94–105, 2018. 36

[37] Jungyub Lee, Sung-Min Oh, and Ju wook Jang. A work in progress: Context based encryption scheme for internet of things. In *FNC/MobiSPC*, 2015. 40, 43

[38] Ruixuan Li, Chenglin Shen, Heng He, Xiwu Gu, Zhiyong Xu, and Cheng-Zhong Xu. A lightweight secure data sharing scheme for mobile cloud computing. *IEEE Transactions on Cloud Computing*, 6:344–357, 2018. 47

[39] Matthew Green, Susan Hohenberger, and Brent Waters. Outsourcing the decryption of abe ciphertexts. In *Proceedings of the 20th USENIX Conference on Security*, SEC'11, page 34, USA, 2011. USENIX Association. 47, 52

[40] Zhibin Zhou and Dijiang Huang. Efficient and secure data storage operations for mobile cloud computing. In *Proceedings of the 8th International Conference on Network and Service Management*, CNSM '12, pages 37–45, Laxenburg, Austria, Austria, 2013. International Federation for Information Processing. 47

[41] L. Touati, Y. Challal, and A. Bouabdallah. C-cp-abe: Cooperative ciphertext policy

attribute-based encryption for the internet of things. In *2014 International Conference on Advanced Networking Distributed Systems and Applications*, pages 64–69, June 2014. 48, 49

[42] Kan Yang and Xiaohua Jia. Expressive, efficient, and revocable data access control for multi-authority cloud storage. *IEEE Transactions on Parallel and Distributed Systems*, 25:1735–1744, 2014. 48, 51, 52

[43] Jin Li, Xinyi Huang, Jingwei Li, Xiaofeng Chen, and Yang Xiang. Securely outsourcing attribute-based encryption with checkability. *IEEE Transactions on Parallel and Distributed Systems*, 25:2201–2210, 2014. 48, 51

[44] Xianping Mao, Junzuo Lai, Qixiang Mei, Kefei Chen, and Jian Weng. Generic and efficient constructions of attribute-based encryption with verifiable outsourced decryption. *IEEE Transactions on Dependable and Secure Computing*, 13:533–546, 2016. 48, 51, 53

[45] Kim Thuat Nguyen, Nouha Oualha, and Maryline Laurent. Securely outsourcing the ciphertext-policy attribute-based encryption. *World Wide Web*, 21(1):169–183, January 2018. 48

[46] Kai Fan, Junxiong Wang, Xin Wang, Hui Li, and Yintang Yang. A secure and verifiable outsourced access control scheme in fog-cloud computing. *Sensors (Basel, Switzerland)*, 17, 2017. 49, 51, 53

[47] Ahmed Saidi, Omar Nouali, and Abdelouahab Amira. Collaborative and fast decryption using fog computing and a hidden access policy. pages 57–71, 11 2019. 49, 50, 51, 62, 103, 106

[48] Cong Zuo, Jun Shao, Guiyi Wei, Mande Xie, and Min Ji. Cca-secure abe with outsourced decryption for fog computing. *Future Generation Computer Systems*, 78:730 – 738, 2018. 49, 51, 53

[49] L. Yeh, P. Chiang, Y. Tsai, and J. Huang. Cloud-based fine-grained health information access control framework for lightweightiot devices with dynamic auditing andattribute revocation. *IEEE Transactions on Cloud Computing*, 6(2):532–544, 2018. 49, 51

[50] Zhidan Li, Wenmin Li, Zhengping Jin, Hua Zhang, and Qiaoyan Wen. An efficient abe scheme with verifiable outsourced encryption and decryption. *IEEE Access*, 7:29023–29037, 2019. 49, 51

[51] Kai Fan, Tingting Liu, K. Zhang, Hui Li, and Yintang Yang. A secure and efficient outsourced computation on data sharing scheme for privacy computing. *J. Parallel Distributed Comput.*, 135:169–176, 2020. 49, 51, 54

[52] Chaosheng Feng, Keping Yu, M. Aloqaily, Mamoun Alazab, Zhihan Lv, and S. Mumtaz. Attribute-based encryption with parallel outsourced decryption for edge intelligent iov. *IEEE Transactions on Vehicular Technology*, 69:13784–13795, 2020. 49, 51, 55

[53] S. Sabitha and M. S. Rajasree. Multi-level on-demand access control for flexible data sharing in cloud. *Cluster Computing*, Nov 2020. 49, 51

[54] Yuanfei Tu, Geng Yang, Jing Wang, and Qingjian Su. A secure, efficient and verifiable multimedia data sharing scheme in fog networking system. *Cluster Computing*, 24(1):225–247, Mar 2021. 50, 51, 54, 106

[55] Kamalakanta Sethi, Ankit Pradhan, and Padmalochan Bera. Pmter-abe: a practical multi-authority cp-abe with traceability, revocation and outsourcing decryption for secure access control in cloud systems. *Cluster Computing*, Jan 2021. 50, 51

[56] Takashi Nishide, Kazuki Yoneyama, and Kazuo Ohta. Attribute-based encryption with partially hidden encryptor-specified access structures. In Steven M. Bellovin,

Rosario Gennaro, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 111–129, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 50, 52

[57] Junzuo Lai, Robert H. Deng, and Yingjiu Li. Fully secure cipertext-policy hiding cp-abe. In Feng Bao and Jian Weng, editors, *Information Security Practice and Experience*, pages 24–39, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. 50, 52

[58] Tran Viet Xuan Phuong, Guomin Yang, and Willy Susilo. Hidden ciphertext policy attribute-based encryption under standard assumptions. *IEEE Transactions on Information Forensics and Security*, 11:35–45, 2016. 50, 52

[59] Lixue Sun and Chunxiang Xu. Hidden policy ciphertext-policy attribute based encryption with conjunctive keyword search. *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pages 1439–1443, 2017. 50, 52, 53

[60] Yinghui Zhang, Dong Zheng, and Robert H. Deng. Security and privacy in smart health: Efficient policy-hiding attribute-based access control. *IEEE Internet of Things Journal*, 5:2130–2145, 2018. 50, 52, 54

[61] Sana Belguith, Nesrine Kaaniche, Maryline Laurent-Maknavicius, Abderrazak Jemai, and Rabah Attia. Phoabe: Securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted iot. *Comput. Networks*, 133:141–156, 2018. 50, 52, 53, 73, 74, 75, 77, 103, 106

[62] Jinmiao Wang and Bo Lang. An efficient and privacy preserving cp-abe scheme for internet-based collaboration. In *CollaborateCom*, 2017. 50, 52, 53, 73, 74, 75, 77, 103, 106

[63] A. A. Abd El-Aziz. An extended data protection model based on cipher-text-policy

attribute based encryption model and an xacml framework in cloud computing. *International Journal of Advanced Science and Technology*, 28(16):1021 – 1033, Dec. 2019. 50, 54

[64] Yang Zhao, Xing Zhang, Xin Xie, Yi Ding, and Sachin Kumar. A verifiable hidden policy cp-abe with decryption testing scheme and its application in vanet. *Transactions on Emerging Telecommunications Technologies*, 2019. 50, 52, 54, 74, 75, 77, 103, 104, 106

[65] Mengting Li, Xinyi Huang, Joseph K. Liu, and Li Xu. Go-abe: Group-oriented attribute-based encryption. In Man Ho Au, Barbara Carminati, and C.-C. Jay Kuo, editors, *Network and System Security*, pages 260–270, Cham, 2014. Springer International Publishing. 51, 52

[66] Yingjie Xue, Kaiping Xue, Na Gai, Jianan Hong, David S. L. Wei, and Peilin Hong. An attribute-based controlled collaborative access control scheme for public cloud storage. *IEEE Transactions on Information Forensics and Security*, 14:2927–2942, 2019. 51, 54, 56, 73, 74, 75, 77, 89, 103, 106

[67] N. Chen, J. Li, Y. Zhang, and Y. Guo. Efficient cp-abe scheme with shared decryption in cloud storage. *IEEE Transactions on Computers*, pages 1–1, 2020. 51, 55

[68] Kan Yang and Xiaohua Jia. Attributed-based access control for multi-authority systems in cloud storage. *2012 IEEE 32nd International Conference on Distributed Computing Systems*, pages 536–545, 2012. 51

[69] Yuanfei Tu, Geng Yang, Junchang Wang, and Qingjian Su. A secure, efficient and verifiable multimedia data sharing scheme in fog networking system. *Cluster Computing*, pages 1 – 23, 2020. 73, 74, 75, 77, 103, 104

[70] Joseph A. Akinyele, Christina Garman, Ian Miers, Matthew W. Pagano, Michael Rushanan, Matthew Green, and Aviel D. Rubin. Charm: a framework for rapidly

prototyping cryptosystems. *Journal of Cryptographic Engineering*, 3:111–128, 2013. 72, 75, 104

[71] G. Ramu. A secure cloud framework to share ehrs using modified cp-abe and the attribute bloom filter. *Education and Information Technologies*, 23:2213–2233, 2018.

[72] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13:422–426, 1970. 94

[73] Bilal Charif and Ali Ismail Awad. Business and government organizations' adoption of cloud computing. In Emilio Corchado, José A. Lozano, Héctor Quintián, and Hujun Yin, editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2014*, pages 492–501, Cham, 2014. Springer International Publishing.

[74] Kaiping Xue and Peilin Hong. A dynamic secure group sharing framework in public cloud computing. *IEEE Transactions on Cloud Computing*, 2:459–470, 2014.

[75] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50âĂŞ58, April 2010.

[76] Ruilong Deng, Rongxing Lu, Chengzhe Lai, and Tom Hao Luan. Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing. *2015 IEEE International Conference on Communications (ICC)*, pages 3909–3914, 2015.

[77] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC âĂŹ12, page 13âĂŞ16, New York, NY, USA, 2012. Association for Computing Machinery.

[78] Shanhe Yi, Cheng Li, and Qun Li. A survey of fog computing: Concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*, Mobidata

âĂŹ15, page 37âĂŞ42, New York, NY, USA, 2015. Association for Computing Machinery.

[79] Ivan Stojmenovic, Sheng Wen, Xinyi Huang, and Hao Luan. An overview of fog computing and its security issues. *Concurr. Comput. Pract. Exp.*, 28:2991–3005, 2016.

[80] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker. Fog computing may help to save energy in cloud computing. *IEEE Journal on Selected Areas in Communications*, 34(5):1728–1739, 2016.

[81] Mohammad M. Bany Taha, Sivadon Chaisiri, and Ryan K. L. Ko. Trusted tamper-evident data provenance. *2015 IEEE Trustcom/BigDataSE/ISPA*, 1:646–653, 2015.

[82] Jungyub Lee, Sungmin Oh, and Ju Wook Jang. A work in progress: Context based encryption scheme for internet of things. *Procedia Computer Science*, 56:271 – 275, 2015. The 10th International Conference on Future Networks and Communications (FNC 2015) / The 12th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2015) Affiliated Workshops.

[83] The architecture of fog network - A bridge between Cloud and IoT, 2020. [online] https://tinyurl.com/4azbzk29.

[84] A.A Abd El-Aziz and Ayman Mohamed Mostafa. A cloud authentication mechanism over encrypted data using xacml framework. *Jokull*, 69(8):7, 2019.

[85] Yong-Woon Hwang and Im-Yeong Lee. A study on cp-abe-based medical data sharing system with key abuse prevention and verifiable outsourcing in the iomt environment. *Sensors (Basel, Switzerland)*, 20, 2020.

[86] Redowan Mahmud and R. Buyya. Fog computing: A taxonomy, survey and future directions. In *Internet of Everything*, 2018.

[87] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K. R. Choo, and M. Dlodlo. From cloud to fog computing: A review and a conceptual live vm migration framework. *IEEE Access*, 5:8284–8300, 2017.

[88] Hadi Zahmatkesh and Fadi Al-Turjman. Fog computing for sustainable smart cities in the iot era: Caching techniques and enabling technologies - an overview. *Sustainable Cities and Society*, 59:102139, 2020.

[89] Naresh Sehgal, Pramod Bhatt, and John Acken. *Cloud Computing with Security: Concepts and Practices*. Springer, 01 2020.

[90] S. Wang, X. Zhang, Y. Zhang, L. Wang, Juwo Yang, and W. Wang. A survey on mobile edge networks: Convergence of computing, caching and communications. *IEEE Access*, 5:6757–6779, 2017.

[91] Liliana Antão, Rui Pinto, João Reis, and G. GonÃğalves. Requirements for testing and validating the industrial internet of things. *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 110–115, 2018.

[92] Ala Al-Fuqaha, M. Guizani, M. Mohammadi, Mohammed Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys and Tutorials*, 17:2347–2376, 2015.

[93] M. Chase. Multi-authority attribute based encryption. In *TCC*, 2007.

[94] Allison Lewko and Brent Waters. Decentralizing attribute-based encryption. In *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, EUROCRYPT'11, page 568âĂŞ588, Berlin, Heidelberg, 2011. Springer-Verlag.

[95] Tim Mather, S. Kumaraswamy, and Shahed Latif. Cloud security and privacy. 2009.

[96] Seongwook Jin, Jeongseob Ahn, Sanghoon Cha, and Jaehyuk Huh. Architectural support for secure virtualization under a vulnerable hypervisor. In *2011 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 272–283, 2011.

ABSTRACT

Today, users are increasingly sharing their data through the cloud; however, the owner loses control when sensitive data isoutsourced. This may result in the exposure of sensitive information without the data owners' consent. This thesis addresses the problem of access control in constrained devices relying on Attribute-based Encryption (ABE); ABE provides many advantages like fine-grained access control. However, it also introduces some challenges. Indeed, these methods present many implementation difficulties because of their complexity and substantial computational and energy overheads. To overcome these issues, we used the benefits of fog computing to create collaborative and distributed versions of ABE schemes. Our methods significantly decrease energy usage and computing overhead. The second limitation of ABE schemes is that the access policy is sent in clear text with the Cipher-text. This could allow a malicious user to compromise the legitimate user's privacy by using sensitive information. We have proposed introducing false attributes, mixed with the real attributes, to preserve the privacy of the access policy. Finally, we tackled the collaboration challenge in the same group among users to decryptdata. Indeed ABE is limited in terms of user collaboration, as they only allow assigning one access authorization to oneuser. To overcome this challenge, we have proposed a collaborative approach that allows users in the same group tocombine their access attributes in a controlled manner to decrypt the data.

Keywords. Attribute Based Encryption, Data Sharing, Decryption Outsourcing, Fog Computing, Collaboration.

Résumé

Aujourd'hui, les utilisateurs partagent de plus en plus leurs données via le Cloud. Cependant, le propriétaire perd le contrôle sur ses données privées lorsque ces dernières sont externalisées au niveau du Cloud. Cela peut entraîner l'exposition d'informations sensibles sans le consentement des propriétaires des données qui est provoqué par un problème au niveau du contrôle d'accès et de sécurité sur ces données. Cette thèse aborde le problème du contrôle d'accès dans le contexte des appareils à ressources limitée et s'appuie sur le chiffrement basé sur les attributs **(ABE)**. En effet, les méthodes ABE présentent de nombreux avantages tels qu'un contrôle d'accès cryptographique précis et fine. Cependant, ces méthodes présentent de nombreuses difficultés de mise en œuvre du fait de leur complexité et de leurs surcoûts de calcul et d'énergie. Pour surmonter ces problèmes, nous avons exploité les avantages du Fog Computing pour créer des versions collaboratives et distribuées des schémas ABE. Nos méthodes réduisent considérablement la consommation d'énergie et la surcharge de calcul. La deuxième limitation des schémas ABE est que la politique d'accès est envoyée en texte clair avec le texte chiffré. Cela pourrait permettre à un utilisateur malveillant de compromettre la confidentialité de l'utilisateur légitime en utilisant les informations sensibles. Pour cela, Nous avons proposé d'introduire de faux attributs, qui sont mélangés avec les attributs réels, pour préserver la confidentialité de la politique d'accès. Enfin, nous avons abordé le problème de la collaboration des utilisateurs au sein du même groupe pour déchiffrer les données. En effet, ABE est limité en termes de collaboration des utilisateurs, car il ne permet d'attribuer qu'une seule autorisation d'accès à un seul utilisateur. Pour surmonter ce défi, nous avons proposé une approche collaborative qui permet à un groupe d'utilisateurs de combiner leurs attributs d'accès de manière contrôlée pour déchiffrer les données.

Mots clés. Chiffrement basé sur les attributs, partage de données, externalisation du décryptage, Fog Computing, collaboration.

ملخص

اليوم، يشارك المستخدمون بياناتهم بشكل متزايد من خلال الحوسبة السحابية. ومع ذلك، عندما يتم الاستعانة بمصادر خارجية للبيانات الحساسة إلى السحابة، يفقد المالك السيطرة على تلك البيانات. فيمكن أن يؤدي هذا إلى الكشف عن معلومات حساسة دون موافقة أصحاب البيانات. تتناول هذه الأطروحة مشكلة access control**في الأجهزة ذات الموارد المحدودة** بالاعتماد على التشفير Attribute-Based Encryption (ABE)، وهو تحد كبير يجب التغلب عليه بشكل مناسب. في الواقع، عندما يتعلق الأمر بإنشاء تحكم دقيق، فإن أساليب **ABE** لها العديد من المزايا. ومع ذلك، فإن هذه الأساليب تحتوي على العديد من الصعوبات في التنفيذ بسبب اعبائها الحسابية والطاقة الكبيرة .للتغلب على هذه المشكلة، استخدمنا مزايا حوسبة الضباب لإنشاء إصدارات تعاونية وموزعة من مخطط **ABE**، طرقنا تقلل بشكل كبير من استخدام الطاقة والاعباء الحسابية. يتمثل القيد الثاني لمخططات **ABE** في إرسال Access **policy** بنص واضح مع النص المشفر. فهاذا قد يسمح لمستخدم ضار بخرق خصوصية المستخدم الشرعي باستخدام معلومات حساسة. للذلك اقترحنا إدخال صفات خاطئة، ممزوجة بالصفات الحقيقية، للحفاظ على خصوصية **access control** حتى لا يتم الكشف عن معلومات مالك البيانات .وفي الاخير، تناولنا تحدي التعاون في نفس المجموعة بين المستخدمين لفك تشفير البيانات، ففي الواقع، **ABE** محدود من حيث تعاون المستخدم، لأنها تسمح فقط بتعيين **access control** واحد لمستخدم واحد. للتغلب على هذا التحدي، اقترحنا نهجًا تعاونيًا يسمح لمجموعة من المستخدمين بدمج صفات الخاصة بهم بطريقة حكيمة ومنضبطة.

الكلمات الدالة. التشفير المستند إلى السمات، ومشاركة البيانات، وفك التشفير، والاستعانة بمصادر خارجية، وحوسبة الضباب، والتعاون.