LIGHTWEIGHT CRYPTOGRAPHIC PROTOCOLS FOR MOBILE DEVICES

A DISSERTATION IN
Computer Science
and
Telecommunications and Computer Networking

Presented to the Faculty of the University
of Missouri–Kansas City in partial fulfillment of
the requirements for the degree

DOCTOR OF PHILOSOPHY

by
ASHWAG OTHMAN ALBAKRI

M.S., University of Pittsburgh, USA, 2015
B. S., King Abdulaziz University, Saudi Arabia, 2010

Kansas City, Missouri
2020

LIGHTWEIGHT CRYPTOGRAPHIC PROTOCOLS FOR MOBILE DEVICES

Ashwag Othman Albakri, Candidate for the Doctor of Philosophy Degree

University of Missouri–Kansas City, 2020

ABSTRACT

In recent years, a wide range of resource-constrained devices have been built and integrated into many networked systems. These devices collect and transfer data over the Internet in order for users to access the data or to control these devices remotely. However, the data also may contain sensitive information such as medical records or credit card numbers. This underscores the importance of protecting potentially sensitive data before it is transferred over the network. To provide security services such as data confidentiality and authentication, these devices must be provided with cryptographic keys to encrypt the data. Designing security schemes for resource-limited devices is a challenging task due to the inherit characteristics of these devices which are limited memory, processing power and battery life. In this dissertation, we propose lightweight polynomial-based cryptographic protocols in three environments that encompass resource-constrained devices which are Wireless Sensor Network (WSN), Fog Computing, and Blockchain Network. With polynomial-based schemes, we guarantee high network connectivity due to

the existence of a shared pairwise key between every pair of nodes in the network. More importantly, the proposed schemes are lightweight which means they exhibit low memory, processing and communication overheads for resource-constrained devices compared with other schemes. The only problem with polynomial-based schemes is that they suffer from node-captured attacks. That is, when an attacker captured a specific number of nodes, the attacker could compromise the security of the whole network. In this dissertation, we propose, for the first time, polynomial-based schemes with probabilistic security in WSNs. That is, when the attacker captured a specific number of sensor nodes, there is a low probability the attacker could compromised the security of the whole network. We show how we can modify system's parameters to lower such attacks.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Graduate Studies, have examined a dissertation titled "Lightweight Cryptographic Protocols for Mobile Devices," presented by Ashwag Othman Albakri, candidate for the Doctor of Philosophy degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Lein Harn, Ph.D., Committee Chair
Department of Computer Science & Electrical Engineering

Deep Medhi, Ph.D.
Department of Computer Science & Electrical Engineering

Vijay Kumar, Ph.D.
Department of Computer Science & Electrical Engineering

Cory Beard, Ph.D.
Department of Computer Science & Electrical Engineering

Sejun Song, Ph.D.
Department of Computer Science & Electrical Engineering

CONTENTS

ILLUSTRATIONS

TABLES

ACKNOWLEDGEMENTS

I am very thankful to my advisor Prof. Lein Harn for his guidance and support through my doctoral research. He is a great advisor who provided me with encouragement, valuable feedback and the expertise that I needed during my dissertation. Without his invaluable support, this dissertation could not have been completed.

My sincere thanks is also extended to Dr. Deep Medhi, Dr.Vijay Kumar, Dr. Sejun Song, and Dr. Cory Beard for serving on my committee and for their time and advice. I express my sincerest gratitude to Dr. Ghulam Chaudhry for his support and advice that gave me continuous strength throughout this entire process.

I would like to thank my parents, Othman Alshehri and Fatima Alshehri, for their love, prayers and endless support. I am grateful to my lovely family for being the source of my strength during my entire academic life. Special thanks go to my husband Mohammed Alshehri who encouraged me to achieve my dreams and helped me while he is pursuing his own doctorate degree. To my beautiful daughter, Diyala and my little son, Mansour, who both have been my steadfast sustainment and motivation. I am also appreciative of my friends for their unconditional support.

Finally, I am indebted to Jazan University for their scholarship and to my country, Saudi Arabia, for funding my education.

CHAPTER 1

INTRODUCTION

The Internet has changed radically from being a small network that used to connect a few computers to a worldwide network that now connects billions of computers around the globe. It was designed to facilitate transferring data and sharing resources between network computers located in different places. When the Internet started, the majority of the connected devices were merely computers. However, in the recent years, different types of devices beyond computers, including smartphones, smart vehicles, and smartwatches have emerged and have the ability to connect and share data. These devices differ from computers in terms of their memory, processing, and battery capabilities. Most of the smart devices (i.e., Internet-of-Things (IoT)) are equipped with sensors that convert physical signals collected from the environment into a digital signal to be processed. These sensors have limited memory, computational capabilities, and low-processing powers but have network capabilities that allow for sharing and exchanging data between devices. For example, cellphones are equipped with GPS sensors by which the data collected can be submitted to the cloud to provide smart navigation services. Another example is sensors that are embedded in a light bulb and allow homeowners to turn lights on and off remotely while managing energy consumption. It is worth mentioning that when transferring data between devices and allowing data to be managed remotely,

these devices need to be connected to the Internet in some way, so their data can be accessed anywhere. Providing such smart services comes with penalties; once the devices are connected to the Internet, they are exposed to various attacks. The proliferation of Internet-of-Things devices necessitates securing the data before transmitting them across networks that involve the Internet.

Why do we care so much about securing the data ? As of today, almost every government, business, institute, and organization in the world is connected to the Internet to provide a convenient way for users to access their online information from their smart devices anywhere and at any time. Nowadays, people can use their smart devices to access their bank accounts, manage their government documents, control their home appliances and access their medical records. This involves submitting sensitive information such as credit cards, social security numbers, or insurance IDs over the network in order to access the services. The Internet is a public channel where various attacks on data exist. The number of data breaches has been increasing dramatically, in which the attackers aim at damaging online services and obtaining sensitive information such as credit cards and medical records. In 2015, a data breach targeted and stole Anthem, Inc.'s servers' results in 37.5 million records containing sensitive information about their customers. Another data breach incident occurred that affected Experian, an information solution company, in which hackers attacked the company server and stole about 15 million T-Mobile customers' records, which included customers credit card numbers and other uniquely identified information [1]. Due to the heavy reliance on technology and the proliferation of Internet-of-Thing devices, it has become increasingly important for

many organizations and companies to *secure online information* and *protect their users' data*; otherwise, the users' information becomes susceptible to attacks.

1.1 What is Security?

The term *'Security'* is defined as "the quality or state of being secure- to be free from danger" [2]; in computing, security means protection against adversaries whose main purpose is to do harm either intentionally or inadvertently. Computer security is a broad term that encompasses, but is not limited to physical, communication, network, and information security where each aims at protecting specific aspects of a computer system. For example, physical security is related to protecting computer hardware, which includes memory, processors, device drivers, etc. Also, communication security focusses on protecting the communications' medias and their contents; while network security provides protection to the networking components and connections. Information security is related to protecting data that are in transit across the network or in storage at various computing devices.

In this dissertation, we refer to the *information security* when discussing the security aspects of various applications.

1.1.1 Security Services

Information security is related to protecting the confidentiality (C), integrity (I) and availability (A) (called CIA-triangle) of the information and data that are either in storage, processing, or transmission [3], the general picture depicted in Figure 1. Each of the aforementioned security services, the CIA-triad, intends to protect a specific attribute

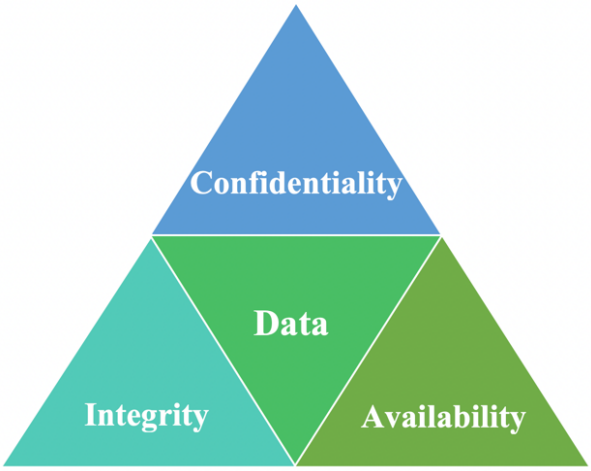

Figure 1: Information Security CIA triangle

of the information, as explained below.

Confidentiality

Confidentiality aims at protecting the information from being accessed by unauthorized parties. Thus, it ensures that only the authorized users have access to the data. Data confidentiality becomes of great importance to protect the personal, financial, health-related information of its users.

Availability

Availability ensures that authorized users can access the information as needed. The information aspect is crucial for many businesses in which the data should be available; otherwise, the unavailability of the data negatively impacts the businesses. For example, online banking makes the data available 24/7 for their customers to access their data; otherwise, unavailable financial services could impact their business and/or personal

accounts. According to Matt Bishop [3], "an unavailable system is at least as bad as no system at al". The availability attribute plays a significant role in critical systems such as power grids, in which the unavailability of such systems could affect the power supplies of the whole nation; redundancy and recovery mechanisms are the main security functions addressing and enhancing the availability of critical computing systems [4].

Integrity

Integrity ensures that information has not been *modified* or *altered* by unauthorized parties. Preserving the integrity and accuracy of information is important in many applications. For instance, in pharmaceutical applications in which the data are very sensitive or where a simple change in one letter could lead to inappropriate medication use or patient harm. In addition, in financial applications, a small change in data could result in having a huge impact on the business. For example, instead of transferring $100,000$ from an account, a slight change in data could result in transferring $1,000,000$ instead, which shows how a small change could have a huge impact and thus making data integrity a significant part of such applications.

Authentication

Authentication is another security service that plays a major role in validating the identity of users in the cyberworld. Authentication aims at ensuring that entities (i.e., a user or system) are who or what they claim to be, and that only the authenticated user or system can access the system and utilize the resources.

1.1.2 Security Methods

In order to provide the aforementioned security services, we need methods or mechanisms. The security method is a tool or procedure designed to provide security services. Cryptography is one set of techniques to provide information security, and it is considered the main building block for many security methods.

Cryptography started as the "art and since of encryption" [5]. Encryption is the main and only tool designed to provide data confidentiality even in the presence of third parties: the adversaries. With encryption, data are protected either in storage or transit around the Internet. That is, if the data has been captured by an adversary, it is considered secured because it is encrypted and thus could not be read. Nowadays, cryptography is a term that involves authentication, hashing techniques, digital signatures, and many more mathematical techniques related to information security.

Digital signatures provide similar properties that are provided by the real-world signatures. It is used to assure a user has approved specific online requests. More importantly, it provides a non-repudiation service that assures a user could not deny a request that involves his/her signature.

A hashing function, also called a message digest, is a one-way function that takes a variable length input and produces a fixed-size output. Hashing functions are utilized in many applications to provide data integrity. For example, malware detection applications check the hash of a file's size to detect if it is embedded with a virus or if it has been modified by malware software. The main use of the hash function is for digital signatures; instead of signing the whole message, which is very large and results in expensive

computational operations, the hash of the message is a small size that results in faster signature processing [5].

In general, the field of cryptography is divided into two main categories: symmetric cryptosystems and asymmetric cryptosystems (Public Key cryptosystems), each of which includes tools and protocols offering security services with different security strengths and requirements. Another key point that is related to the cryptosystems is key management.

1.2 What is the Key Management?

In order for two parties to communicate and exchange data securely across networks, they must share a key in advance before transmitting any data. The key that is shared between the parties must be delivered through a secure channel; otherwise, if an adversary captures the key, he/she could decrypt all data transferred between the parties. Due to the significant role of keys in securing data, key management is a crucial part in many computing systems. Key management covering all aspects related to key generation, distribution, establishment, and revocation.

In symmetric cryptosystems, the same key is used for both encrypting and decrypting data. That key represents a shared secret between two or more parties who want to share secret information. If the secret key is compromised, all data related to all parties are compromised as well. Thus, the secret key must be revoked, and new keys must be generated and distributed securely to all parties. Due to the shared access to the cryptographic key, the key management becomes complicated, especially when the number of

parties increase.

On the other hand, in asymmetric cryptosystems, every party in the system has a pair of keys: a public and private key. The public key can be seen by anyone in the system, while the private key is kept in a secure place and only known to its owner. The public key and private key are correlated in some way, but it is computationally infeasible to drive the private key from the public key [6]. When the data are encrypted with a public key, they can be decrypted with the corresponding private key and vice versa. If the private key of one party gets compromised, only data related to that party are affected. In addition, only the compromised party needs to get a new pair of keys. The public-Key Infrastructure (PKI) makes the key management simpler, and it is an efficient scheme, especially in large networks. The PKI involves a certificate authority (CA) that is responsible to create digital certificates for every entity in the network. The digital certificate binds the party's identity to its public key, which is used to authenticate valid parties and eliminate any adversary that tries to impersonate a valid party in the system. Although both symmetric and asymmetric cryptosystems provide encryption and authentication methods, an asymmetric cryptosystem is the only one to provide digital signatures and non-repudiation security services. Because each party has its own private key, which is only known to its owner, the encryption with the private key is called the signature of the owner of that private key. Anyone with an access to a party's public-key can use it to validate its signature. With this feature, a public-key cryptosystem provides a non-repudiation security mechanism that assures a party could not deny the validity of doing something since it includes its own signature.

However, symmetric cryptosystems are very fast compared with public-key cryptosystems and makes them suitable for encrypting large messages. Symmetric cryptosystems require smaller key sizes and a smaller number of computations and result in a fast processing (i.e., encryption or decryption) of data. However, because the security of asymmetric cryptosystems is based on some computational assumptions, it requires large key sizes, which results in more computations that make their data processing very slow when compared with the symmetric cryptosystems. For the aforementioned reasons, asymmetric cryptosystems are mainly utilized for key management and digital signatures, while symmetric cryptosystem are utilized for encryption/decryptions of data [6].

1.2.1 What is Key Distribution?

Key distribution is the main component of security subsystems of both distributed systems and communication networks [7]. The proliferation of devices that are smaller in size and limited in their memory, processing, and battery capabilities, which become an integral part of the network systems, pose new challenges and require novel key distribution approaches.

The existing cryptographic protocols for key distribution schemes in computer networks are not feasible for resource-constrained devices due to the devices' inherent characteristics (i.e., limited memory, processing, and battery powers). Public-key infrastructure has been utilized for key distribution (i.e., key exchange protocols) in computer networks. However, public-key cryptosystems are not suitable for resource-constrained

devices because their memory and processing requirements exceed the devices' capabilities. There are many approaches introduced in the literature addressing key distribution in systems that are comprised of limited-resource devices.

In this work, we aim at providing lightweight cryptographic solutions for key distribution/management in various environments that encompass constrained devices.

1.3 Application Domains

There are a great number of applications such as healthcare and military applications in which security is an inherent part of the services provided by such applications. In this dissertation, we consider three popular application domains: Wireless Sensor Networks (WSN), Fog Computing, and Blockchain Networks.

1.3.1 Wireless Sensor Networks (WSN)

A Wireless Sensor Network (WSN) is an infrastructure comprised of sensing, computing, and communication entities called sensors, which provide the ability to observe and react to phenomena in a specific environment that can be the physical world, a biological system, or an information technology framework [8]. WSNs have been utilized for data acquisition, monitoring, and biotelemetry purposes. Due to the sensitivity of data transferred over the WSN, it is important to preserve the confidentiality of data from being compromised by attackers. For that reason, security services such as data confidentiality and authentication are required to secure WSNs. However, securing the WSNs is a challenging task due to the inherent characteristics of the sensors (limited memory, processing, and power capabilities). In this work, we aim at designing lightweight key

management protocols in WSNs by taking into account the physical limitation of sensor devices.

1.3.2 Fog Computing

With the proliferation of smart devices (i.e., IoT), that are equipped with Internet-connectivity capabilities, several concerns have been raised related to network connectivity and security. The smart devices are connected to the cloud and provide a way to manage, process, and control devices remotely both anywhere and at any time. Recently, it is anticipated that 50 billion IoT devices will be connected to the cloud by 2020 [9]. That rapid increase of smart devices connected to the cloud creates several network issues such as high latency and bandwidth congestion, which are not suitable for many delay-sensitive applications. To resolve the aforementioned issues, fog computing has been proposed and is a paradigm that extends the cloud to the edge of the network [10]. With fog computing, several benefits are provided such as low latency, location awareness, mobility, and the adoption of a large number of heterogeneous devices [11]. Despite the network solutions provided by fog computing, security concerns have not been fully considered thus far. The security schemes applied for the cloud could not implemented directly to fog nodes. In this work, we demonstrate security challenges in a fog network and address the security concerns related to preserving confidentiality of the data produced by smart devices and transferred over fog networks. Specifically, we propose a key management scheme in fog computing that facilitates key distribution and protects data transferred across networks.

1.3.3 Blockchain Network

Lately, *Blockchain* has received a lot of attention from many people in research and financial institutions, industries, and governments. It is claimed that the blockchain is the biggest technological invention since the emergence of the Internet [12]. After the invention of the first cryptocurrency, *Bitcoin*, blockchain received a lot of attention due to its key features such as decentralization, anonymity, persistency, and auditability [13]. Although Bitcoin is the most popular blockchain application, blockchain technology can be applied into various applications beyond cryptocurrency such as healthcare [14], vehicular networks [15] and smart homes [16]. Blockchain is a distributed public ledger in which all transactions are stored in blocks, and all the blocks are chained together using cryptographic mechanisms. At the time when many of the transactions took place through a trusted third party, blockchain provided a distributed system of trust in a peer-to-peer networks in which entities of the networks handled the trust of the transactions in the network and thereby eliminated the need for a third party. In this work, we investigate the security aspects of the blockchain technology; more specifically, we scrutinize the network structure and propose a key distribution scheme that facilitates a key establishment between entities in the network.

1.4 Our Contribution

Today, network deployment includes resource-constrained devices that are limited in their memory, computational, and battery power. To protect data transferred over the

network, entities must encrypt data before transmitting them over the network. To encrypt data, keys must be distributed securely to each entity in the network that is used to encrypt the data. There are two encryption schemes: symmetric cryptosystems and public-key cryptosystems. Although a public-key cryptosystem does not need key distribution, it requires a large amount of memory to store keys and complex computations to encrypt/decrypt data that are not suitable for a network that involves resource-constrained devices such as sensors and IoTs. The suitable encryption scheme for such networks is the symmetric cryptosystem because it has less storage requirements and a smaller number of computations compared to the public-key schemes. However, the symmetric cryptosystem needs a key distribution scheme to securely distribute secret keys to all entities in the network.

In this dissertation, three polynomial-based key distribution schemes in three environments have been proposed. They are the Wireless Sensor Networks (WSN), Fog Computing, and Blockchain Networks. Our contribution is twofold. First, we designed *lightweight* key distribution schemes, which means our schemes require less memory, computations and battery life compared with other schemes applied in WSN, Fog networks, and permissioned Blockchain Networks. Secondly, we proposed, for the first time, a polynomial-based scheme with *probabilistic* security in WSNs. All polynomial-based schemes are deterministic security, which means capturing a specific number of nodes in the network deterministically leads to a compromise of the whole network's security. With probabilistic security, we enhance the security of the network by decreasing the probability of successful node-captured attacks.

We summarize the contribution of this dissertation work as follows:

- In WSNs, we propose a novel deterministic-key distribution scheme with probabilistic security. We guarantee a shared key exists between every pair of entities in the network. More importantly, with a probabilistic security feature, there is a low probability for sensor- captured attacks to successfully reconstruct the polynomials that are used to generate tokens for all nodes in the network. We aim at reducing sensor-captured attacks as much as possible. We also show that we can adjust a system's parameters to lessen such attacks and enhance the security of the network.

- In Fog Computing, we noticed that most of the current schemes are either based on the Public-key Infrastructure (PKI) or required high communication and computation overhead. We propose a lightweight polynomial-based key distribution scheme in Fog Networks. Our scheme exhibits low storage, computation, and processing requirements, which make it suitable for fog networks that encompass resource-constrained devices (IoTs).

- In Blockchain networks, we noticed that most of the blockchain implementations are based on the Public-Key Infrastructure (PKI). The PKI requires more resources to send and validate transactions in the blockchain network. In this work, we propose a polynomial-based key management scheme in the blockchain network, more specifically, we consider the permissioned blockchain. The main goal is to eliminate the PKI overhead by preloaded entities with tokens that facilitate the establishment of shared keys and transmission of data securely within the network. As a

result, this reduces the time to *process* transactions and *transfer* data.

1.5 Dissertation Arrangement

The remainder of this dissertation is organized as follows. In Chapter 2, we demonstrate cryptographic protocols as well as cryptographic tools utilized as the basis in our proposed solutions. Chapter 3 discusses the related work in WSN, Fog Computing and Blockchain Network. In Chapter 4, we introduce the Wireless Sensor Networks (WSNs) with the security services required in such networked systems. Also, we propose two lightweight cryptographic solutions in WSNs for key establishment and group communication. In Chapter 5, we elaborate about fog computing and the security services required in fog networks. In addition, we extend Chapter 5 to include our hierarchical key management scheme in fog computing as well as the security analysis of the proposed scheme. Chapter 6 encompasses blockchain network, more specifically, Hyperledger fabric, the permissioned blockchain, the system structure, network communications, and security services provided for entities in the network. We propose a novel key management structure in the permissioned blockchain coupled with a security analysis of the proposed scheme and a comparison with the current scheme used in the Hyperledger Fabric. Finally, we conclude this dissertation and provide some pointers to future research work in Chapter 7.

CHAPTER 2

OVERVIEW OF CRYPTOGRAPHICAL KEY DISTRIBUTION SCHEMES

In this chapter, we provide an overview of cryptography and demonstrate cryptographic services and techniques to provide those services. In addition, we discuss various key distribution schemes that are the main focus of this dissertation work. More specifically, we focus on polynomial-based key distribution schemes that are utilized in designing our proposed schemes in the following chapters.

2.1 Overview of Cryptography

The goal of cryptography is to provide an efficient way for people to communicate securely over a public channel in which an adversary who eavesdrops on the channel's communications cannot understand what is being sent [6]. In the past, cryptography referred to a means of encryption that aimed at preserving a message's confidentiality. With encryption, the message is transformed into an unreadable format, so adversaries and eavesdroppers cannot recover the original message without secrete knowledge. At that time, the main objective of cryptography was to ensure secrecy in communications. However, in recent years, cryptography has expanded and now encompasses various techniques for ensuring message integrity, identity authentication, digital signature, and non-repudiation.

In general, the cryptography field can be divided into two broad categories: Symmetric Cryptosystems and Asymmetric Cryptosystems. Before diving into the cryptosystems, let us explain some terminologies that are used in the description of the encryption schemes.

The original message is called *plaintext*, which is used as input to the encryption algorithm. Also, a *key* is used as input to the encryption algorithm to transform the plaintext into an unreadable format, called *ciphertext*. An adversary could observe the ciphertexts transferred through the communication channel but cannot recover the plaintext from the ciphertext without the knowledge of the key, which must be kept secret by the communicating parties. There are several network entities; we refer to the communicating parties as *Alice* and *Bob*, and the adversaries as *Oscar* or *Eve* who eavesdrop on the communication channel and try to know the plaintext of the captured ciphertext or obtain the key used to secure the transmitted data.

2.1.1 Symmetric Cryptosystem

The symmetric cryptosystem was the only cryptosystem available and utilized prior to the development of its counterpart, the asymmetric cryptosystems, in the 1970s [17]. The symmetric cryptosystems are also called, conventional encryption or one-key encryption systems. In a symmetric cryptosystem, Alice must pre-share a key with Bob in a *secure* fashion prior to submitting any data across the network, as depicted in Figure 2. Alice inputs both the key and the plaintext into the encryption algorithm, which outputs the ciphertext that can be transferred over the network to Bob. Bob, on the other hand, uses

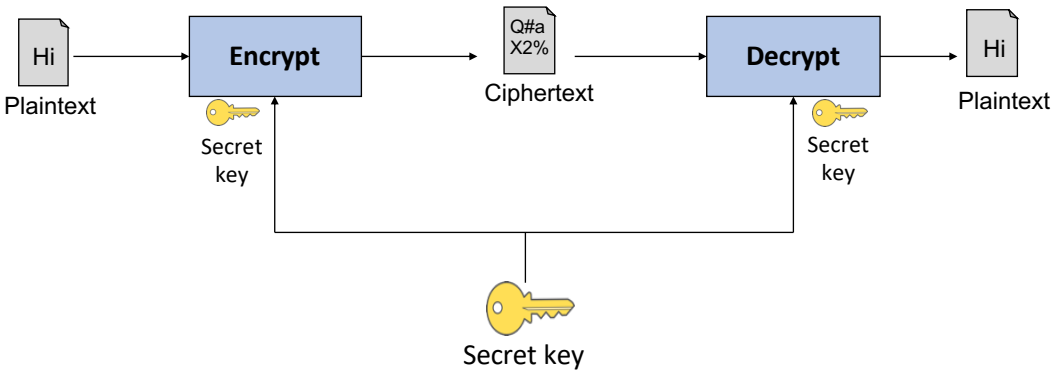

Figure 2: Symmetric Cryptosystem

the key shared with Alice, along with the ciphertext, as inputs to the decryption algorithm to produce the plaintext. In the same way, Oscar intercepts the communication channel to obtain the ciphertext; however, he cannot recover the plaintext from the ciphertext without knowing the key.

One of the drawbacks of the symmetric cryptosystem is that once the key is compromised the whole communication and all data transmitted between Alice and Bob are compromised, too. That necessitates regenerating a new key and distributing it securely between the Alice and Bob entities in the network.

2.1.2 Asymmetric Cryptosystem

In the asymmetric cryptosystems, each entity in the network has two keys, a public kay and a private key. Encrypting a message with a public key requires using the corresponding private key to decrypt the message, and vice versa, as shown in Figure 3. Asymmetric cryptosystems are also known as public-key or two-key encryption cryptosystems. The public key can be openly distributed or published in an open directory,

while the private key must be kept secure and only known to its owner. Although the public and private keys are mathematically related, it is computationally infeasible to derive the private key from the public key [17]. With a public-key cryptosystem, Alice can send a message securely to Bob using Bob's public key, which can be found on a public directory; on the other hand, only Bob, who owns the private key, can decrypt the ciphertext and recover the original message. It is computationally infeasible for Oscar, the adversary, knowing the ciphertext and Bob's public key, to recover the original message. It is also computationally infeasible for Oscar, knowing a public key, to determine the corresponding private key [17].

If a private key of an entity gets compromised, then only the compromised entity needs to regenerate a new key. Thus, compromising a private key of an entity only affects the compromised entity and does not necessitate other non-compromised entities to get a new key pair. On the contrary, if a key, which is shared between two (or more) entities gets compromised, a symmetric cryptosystem needs to regenerate a new key and distribute it securely to the entities.

2.1.3 Comparison between Symmetric and Asymmetric Schemes

The one problem associated with all symmetric cryptosystems is that a key must be shared between entities through a *secure* channel prior to transmitting any data over the network. Moreover, in a large network, the number of keys required to secure communications between entities is large and grows when a new entity joins the network. For instance, if the total number of entities in the network is n, then the total number of keys

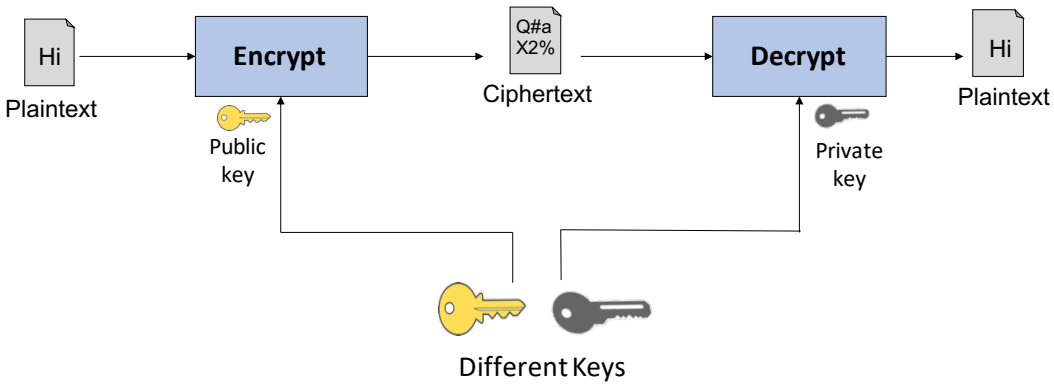

Figure 3: Asymmetric Cryptosystem

shared between every pair of entities in the network is $n * (n-1)/2 = O(n^2)$. When the network size increases, the number of keys increases as well, which complicates the key management process which involves establishing, distributing, revoking, and updating the keys. Thus, the aforementioned key distribution is complicated and impractical for large networks.

On the contrary, the public-key cryptosystems provide a way for network entities to exchange messages *securely* without the need of prior communication of a secret key through a secure channel. The public-key cryptosystems rely on the public-key infrastructure (PKI) for key management purposes. The PKI includes a certificate authority (CA), which is a trusted authority that is responsible for generating certificates for the entities in the network. The certificate binds the public key with the identity of the user, and it is signed by the CA. Thus, anyone can validate the certificate by checking the signature of the CA. When entities want to communicate, they need to exchange their certificates and validate the CA's signature, then get the public keys from the certificate. Because the

public key (in the certificate) is available in an open directory, it can be used to *securely* exchange a message or a key of a symmetric cryptosystem over an *insecure* channel. The problem of using public-key cryptosystems is that encrypting a message with a public or a private key requires a large number of computations that not only slows the encryption process but also produces large encrypted messages compared to the symmetric cryptosystems. Thus, public-key cryptosystems are mainly utilized for key exchange protocols while the symmetric cryptosystems are still used for message encryption. The key exchange protocol provides a way for two entities to cooperate in establishing a session key, which is a key shared between the two entities and generated for a particular session. The session key is utilized in a symmetric encryption scheme, and it is valid during a short period of time [17]. Furthermore, public-key cryptosystems outweigh symmetric cryptosystems for exclusively providing additional security services such as digital signature and non-repudiation, in which Alice (i.e., the sender) uses her private key to encrypt a message producing what is called a *digital signature*, and Bob (i.e., the receiver) uses Alice's public key to decrypt the message. This provides proof that Alice is the one who signs the message. In addition, Alice cannot deny the submission of a signed message since it is encrypted with her private key to which only she has access.

Table 1 summarizes the advantages and disadvantages of symmetric and asymmetric cryptosystems. From encryption's perspective, symmetric cryptosystems require smaller key sizes compared to asymmetric cryptosystems. For example, the minimum key size for an AES encryption scheme is a 128-bit key size while the RSA requires at least 1024 [18]. Additionally, the symmetric encryption scheme is very fast compared

Table 1: Comparison between Symmetric and Asymmetric Cryptosystems

Criteria	Advantage	Disadvantage
Symmetric cryptosystem	Small key sizes Fast encryption	Complex key management schemes
Asymmetric cryptosystem	Key exchange. Digital signature Public-Key Infrastructure (PKI)	Large key sizes Slow encryption

to the public-key schemes. Due to the aforementioned features of symmetric encryption, the symmetric cryptosystem is utilized to encrypt the messages while public-key cryptosystems are used for the key exchange protocol and to provide digital signature security services. Again, the drawback of the symmetric cryptosystem is that the key management scheme is complex compared to the public-key schemes, which have a Public-Key Infrastructure (PKI) that facilitates key management processes.

2.2 Key Management Schemes

Key management processes all the management aspects of the cryptographic keys in a cryptosystem. They encompass operations related to key generation, distribution, storage, derivation, revocation, and replacement of the compromised keys [19]. Designing a secure key management scheme is crucial since it has a great impact on the security of the whole system. A weak key management scheme leads to an inefficient and vulnerable system. There are several factors that need to be taken into account when designing a secure key management scheme such as the type of the application in which the cryptographic keys are used and possible threats on the systems.

This dissertation focuses on the key distribution part of key management. As mentioned in Chapter 1, key management, specifically, key establishment and key distribution are takin into consideration in three environments: Wireless Sensor Networks (WSN), Fog Computing, and the Blockchain Network. Each is explained in detail in the next chapters.

2.2.1 Classification of Key Distribution Schemes

In this section, several key distribution approaches are discussed. Key distribution schemes are classified into two main categories: centralized schemes and non-centralized schemes, as depicted in Figure 4.

Before discussing the different key distribution approaches, we must differentiate between two types of keys: a long-lived key (i.e., a master key or secret key) and a short-lived key (i.e., a session key). The long-lived keys are generated and distributed to the network's entities through a *secure* channel. Usually, they are fixed and not changed during the network's lifetime, which necessitates that they be kept secure; otherwise, it would affect the security of the network. In practice, long-lived keys are used to encrypt session keys that can be sent over an *insecure* channel. On the other hand, short-lived keys are generated to secure the communication between the network's entities during a short period of time (i.e., session). Those keys are temporary keys, which are used during a session, and then discarded once the session ends [6].

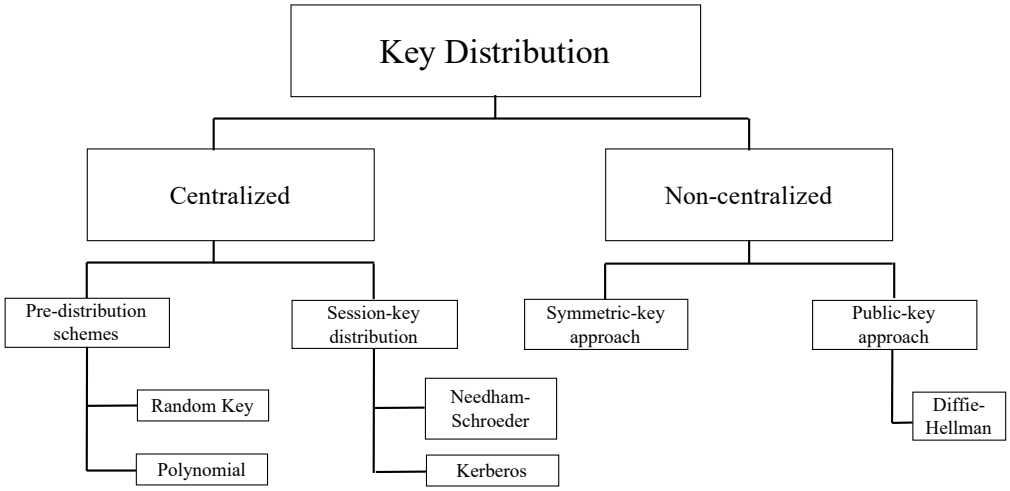

Figure 4: Key Distribution Schemes

2.2.1.1 Centralized Key Distribution Schemes

In the centralized schemes, a *key distribution center (KDC)*, also called a *key server*, plays the main role of generating secret keys and distributing them over a *secure* channel. Initially, each entity must share a unique secret key (i.e., a master key) with a KDC for the goal of key distribution. Then, if two entities want to communicate securely, they request a session key from the KDC that creates a session key and encrypts it with the secret keys shared between the KDC and the network's entities. The centralized key distribution schemes are divided into *pre-distribution schemes*, which are designed to facilitate the distribution of the secret keys, and *session-key distribution schemes*, which relay on the centralized key server (i.e., KDC) to generate and distribute the session keys.

(a) Pre-distribution Schemes

In the pre-distribution scheme, a KDC is responsible for generating keying materials and distributing them *securely* to all entities in the network. Later, if two entities want to communicate, they use the preloaded keying information to establish a key that is utilized to secure data transferred between the entities. We demonstrate two widely used pre-distribution schemes: The *Random Key* scheme and *Polynomial-based* scheme.

- Random Key Scheme

  In the random key scheme [20], the KDC generates a large pool of keys and randomly draws a subset of keys from the pool and loads it into the network entity's storage, which is called the *key ring*. Thus, each entity in the network is preloaded with a subset of keys in its key ring. That is the first step, which must be done *securely* before network deployment. Later, when two entities want to communicate, they need to search their key rings to find at least a common key that can be used to secure data transferred between entities. That is called *network connectivity*. If there is no common key found in the entities' key rings, the entities start to search for a neighbor who has common keys between the entities and is able to establish a key and share it with the entities. That is called *path discovery*. As we see from the aforementioned scenario, the random key approach does not guarantee that a common key exists between every pair of entities in the network and that is why it is called a *probabilistic* scheme. To increase the probability that two entities share at least a common key, we need to increase the size of the key rings, which not only increases the storage requirement of the entities, but also increases the vulnerability against node-captured attacks. A *node-captured attack* is an attack in

which the attacker physically captures a node or virtually attacks the node to obtain the secret data (i.e., keys or keying materials) that are stored in the nodes. When an attacker captures more entities, more keys from the key pool are revealed and this leads to compromising more network links of other uncompromised entities. The random-key scheme is a *probabilistic* security scheme. This means it is probabilistic to compromise the network security when a specific number of entities' key rings are compromised by attackers. Note that it is possible that the same key is shared by more than a pair of entities in the network, since the key rings consist of keys drawn from the same key pool. As a result, there is always a tradeoff between network connectivity and resistance against node captured attacks.

- Polynomials-based Scheme

  A polynomial-based key distribution scheme aims at allowing two entities to establish a key independently and use it to encrypt data transferred over the network [21]. In a polynomial-based scheme, each entity in the network is assigned a unique identity (ID). A KDC creates a polynomial; and then for each entity in the network, the KDC evaluates the polynomial at the entity's identity and loads the polynomial's coefficients (i.e., keying material) into the entity's storage. Later, when two entities want to communicate, they utilize the stored polynomial's coefficients with the entities' identities to *independently* create a key to secure their data. The polynomial-based scheme is called a *deterministic* scheme because it guarantees that a key exists and is shared between every pair of entities in the network. Although the

polynomial-based scheme exhibits high network connectivity, it suffers from node-captured attacks. It is also a *deterministic* security scheme, which means that when an attacker captures a specific number of nodes, the attacker is *deterministically* able to recreate the polynomial utilized by the KDC and regenerate all the keying materials (i.e., polynomial's coefficients) for all nodes in the network. This results in compromising the security of the whole network. To reduce the node-captured attacks, we need to increase the degree of the polynomial, which is the security parameter and the threshold of the scheme. However, increasing the degree of the polynomial results in increasing both the storage space and number of computations required by the network's entities. We demonstrate the structure and the characteristics of polynomial-based schemes in detail in Section 2.3.

Table 2 compares polynomial-based schemes with the random key schemes. Random key schemes are probabilistic schemes that do not guarantee a common key exists between every pair of entities in the network, and thus results in a partially connected network. On the other hand, polynomial-based schemes are deterministic schemes that guarantee a key exists and is shared between every pair of entities. This generates a fully connected network. In terms of storage, random key schemes load each entity with a key ring of a specific size, while polynomial-based schemes load entities with polynomial coefficients that are related to the degree of the polynomial. In terms of computations, random key schemes do not require any computation to establish the keys because the keys are already loaded in the key rings;

Table 2: Comparison between Random Key and Polynomial-based Schemes

Comparison Criteria	Random Key Schemes	Polynomial-based Schemes
Network Connectivity	Partially connected (Probabilistic)	Fully connected (Deterministic)
Storage Requirement	Key Ring	Polynomial Coefficients
Computation Requirement	-	Polynomial Evaluations
Communication Overhead	High	Low
Security Against Node-Captured Attacks	Probabilistic	Deterministic

however, entities that utilize a polynomial-based scheme need to evaluate the polynomial to establish the keys. In terms of communication, the messages exchanged between entities in order to find a common key or run a path discover phase is high, while entities in the polynomial-based schemes only need to exchange their IDs. Finally, the security of random-key schemes against node-captured attacks is probabilistic, while polynomial-based schemes are deterministic security schemes.

(b) Session Key Distribution Schemes

The centralized session key scheme assumes that secret keys between the KDC and network's entities are already distributed securely (i.e., through the pre-distribution phase) and loaded in the entities' storage. We start to explain two session key schemes: *Needham-Schroder* and *Kerberos*.

- Needham-Schroder

In the Needham-Schroder protocol [22], when Alice wants to send data securely to Bob, she first sends a request to a KDC to generate a session key and send it to herself , which in turn, sends it securely to Bob. The protocol is depicted in Figure 5. From the Figure, Alice submits her request for a session key to the KDC, which includes a nonce (i.e., a random number r_{A1}), her identity (ID_A), and Bob's Identity (ID_B). Then, the KDC sends an encrypted reply message by the secret key ($K_{A,KDC}$), which includes the nonce (r_{A1}), Bob's identity (ID_B), the session key ($K_{A,B}$), and a message encrypted with a secret key between the KDC and Bob ($K_{B,KDC}$). This includes the session key ($K_{A,B}$) and Alice's identity (ID_A). Alice then decrypts the message with her secret key ($K_{A,KDC}$), then forwards the encrypted message with Bob and KDC secret key along with a nonce (r_{A2}) that is encrypted with the session key, ($K_{A,B}(r_{A2})$). After that, Bob receives the message, decrypts it with his secret key shared with the KDC and retrieves the session key, ($K_{B,KDC}(ID_A, K_{A,B})$). Then, Bob uses the received session key ($K_{A,B}$) to decrypt the second message ($K_{A,B}(r_{A2})$) and retrieve the random number (r_{A2}). Bob then uses the session key shared with Alice ($K_{A,B}$) to encrypt a message which includes a new nonce (r_B), and Alice's nonce, after doing some calculations, (i.e., $r_{A2} - 1$). By decrypting the message with the session key $K_{A,B}$ and getting the nonce, Alice authenticates Bob and makes sure that he has received the session key and her nonce. Finally, Alice encrypts Bob's nonce and do some calculations (i.e., $r_B - 1$), which authenticates Alice to Bob and shows that she is able to decrypt his previous messages and did receive his nonce.

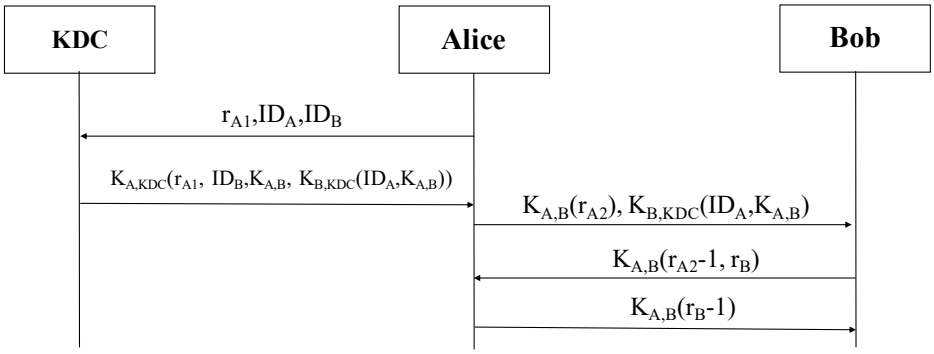

Figure 5: Needham-Schroeder Protocol

- Kerberos

  The above Needham-Schroeder protocol is vulnerable to the replay attack in which an attacker uses a previous compromised session key ($K_{A,B}$), along with the message submitted previously to Bob ($K_{B,KDC}(ID_A, K_{A,B})$), in order to fool Bob who accepts the message and can not tell that it was an old one [17]. To fix that problem and eliminate reply attacks, Kerberos introduced the timestamp in the protocol [23]. When the timestamp expired, it indicated that the session key expired too. Figure 6 illustrates the Kerberos protocol, which works as follows:

  - Alice chooses a random number (r_A) and sends it along with her ID ($ID(A)$) and Bob's ID ($ID(B)$) to the KDC.
  - The KDC chooses a random session key (K) and timestamps it ($timestamp$). Then, it prepares a ticket message for Bob, $t_{Bob} = e_{K-Bob}(K||ID(A)||timestamp)$, and another message $y_1 = e_{K-Alice}(r_A||K||ID(B)||timestamp)$, where both t_{Bob} and y_1 are sent to Alice.

- Alice decrypts y_1 using her secret key, $K - Alice$ and gets the session key, K. Next, Alice computes the current time ($time$) and prepares y_2, $y_2 = e_K(ID(A)||time)$ and then she sends t_{Bob} and y_2 to Bob.
- Once Bob receives the messages from Alice, he decrypts t_{Bob} using his secret key $K - Bob$ to obtain the session key, K. He then uses the session key to decrypt y_2 to obtain $time$. Bob next computes: $y_3 = e_K(time + 1)$ and sends it to Alice.
- Finally, Alice decrypts the y_3 and checks to see if Bob did the correct computation.

One of the drawbacks of Kerberos is that all network entities must have *synchronized clocks* to determine the current time (i.e., $time$) used to validate the session keys. Because ensuring a perfect synchronization is very difficult in practice, we should consider a slight amount of variation in time [6].

2.2.1.2 Non-Centralized Key Distribution Schemes

In non-centralized schemes, the two entities who want to communicate, engage directly in generating and establishing session keys without involving a third party such as a KDC. We explain two non-centralized schemes to establish session keys: a *symmetric-key* approach and a *public-key* approach.

(a) Symmetric-key Approach

Two network entities cooperate to create a session key by utilizing their secret keys. Every entity in the network shares secret keys with all other entities, which results

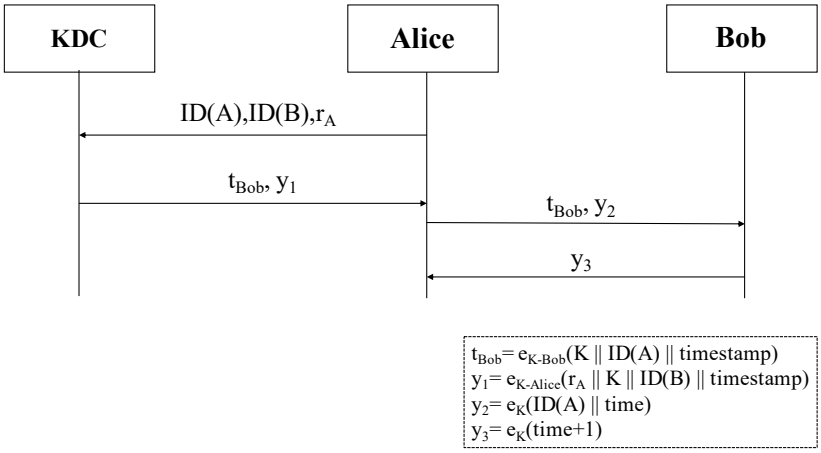

Figure 6: Kerberos

in each entity storing $n - 1$ keys. To establish a session key, one of the entities who initiates the communication, Alice, sends a request to the responder, Bob, asking for a session key to be used to secure the data. Bob creates a random session key and encrypts it under the secret key shared with Alice. The symmetric-key approach is shown in Figure 7. First, Alice sends her ID ,$ID(A)$, and a nonce, r_A, to Bob. Bob then generates a session key (K_s) and sends it along with Alice's ID, $ID(A)$, his ID, $ID(B)$, and a new nonce (r_b). He next runs a function on Alice nonce($f(r_A)$) and all are encrypted with the secret key shared with Alice (K), $e_K(K_s||ID(A)||ID(B)||f(r_A)||r_b)$. Once Alice receives a message from Bob, she decrypts it with the secret key, obtains the session key, and validates the function on her nonce. Finally, Alice submits a message that is a function of Bob's nonce, $f(r_b)$, that is encrypted with the session key, $e_{K_s}(f(r_b))$, [17]. Bob then ensures that Alice got the session key and that she is calculating the correct nonce and not replying to a previous message.

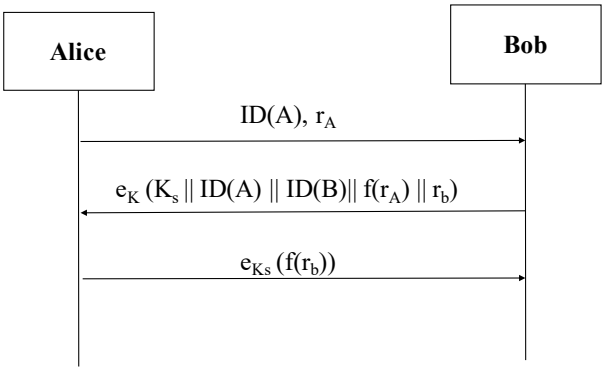

Figure 7: Symmetric Approach for Session-key Distribution

(b) Public-key Approach

Every entity in the network has a keypair, public and private keys, which can be used for symmetric key distribution. One of the well-known key exchange protocols is the *Diffie-Hellman* Key Pre-distribution Scheme [24]. This protocol is illustrated in Figure 8, and it works as follows:

- The public domain parameters are a group $(G, .)$ and an element $\alpha \in G$ with an order n
- Alice chooses a private key (a_{Alice}), and generates her public key as $KU_A = \alpha^{(a_{Alice})}$.
- Alice uses Bob's public key (KU_b) to generate the secret key as $K_{A,B} = (KU_b)^{a_{Alice}} = \alpha^{(a_{Bob} * a_{Alice})}$
- In the same way, Bob chooses a private key (a_{Bob}) and generates his public key as $KU_B = \alpha^{(a_{Bob})}$

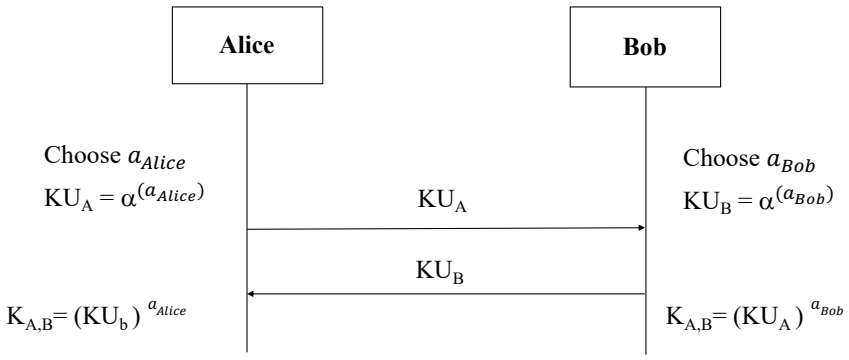

Figure 8: Public-key Distribution Scheme

- Bob uses Alice's public key (KU_A) to generate the secret key as $K_{A,B} = (KU_A)^{a_{Bob}}$ $= \alpha^{(a_{Alice} * a_{Bob})}$

Thus, at the end, Alice and Bob share the same key that can be used to secure data transmitted over the network. The problem in the above scheme is that it is vulnerable to the Man-in-The-Middle Attack (MIMA) because the protocol lack the authentication of the communicating parties [6].

Table 3 summarizes the features of centralized and non-centralized key distribution schemes. Centralized schemes are implemented widely across several application domains as they exhibit a fast way to distribute keys compared with non-centralized approaches. However, centralized approaches require interaction with a third party, such as KDC or CA, which must be available all the time to respond to key distribution requests. Some centralized approaches utilize a $timestamp$ that requires clock synchronization between entities in the network. With large networks, the centralized approach suffers from a bottleneck when trying to respond to a high number of keys' requests from all nodes in the networks. The main advantage of utilizing a non-centralized approach is that when

Table 3: Comparison between Centralized and Non-centralized Key Distribution Schemes

	Centralized Schemes	Non-centralized Schemes
Advantage	Fast and Practical	No need for 3rd party
Disadvantage	Require 3rd party Require clock synchronization 3rd party must be available all time Bottleneck (with large network)	Require pre-distribution schemes

entities want to communicate, they directly interact with each other to establish a session key without the need of a third party; however, it is based on a pre-distribution scheme in which secret keys (or certificates) are distributed in advance between entities in the network through a KDC or a CA.

2.3 Polynomial-based Key Distribution Schemes

In this section, we demonstrate, in detail, the characteristics of the polynomial-based key distribution schemes, the topic of this dissertation.

In 1983, Blom [21], [25] introduced a novel key distribution scheme for symmetric cryptosystems based on polynomials which were utilized by a KDC as a tool to generate keying materials for every user in the network. The KDC distributed the keying materials securely to every user in the network. Later, each user could use the stored keying material to generate secret keys shared with other users in the network. The proposed scheme [21], [25] guaranteed two features: low storage requirements to store a keying material, and greater theoretical security than public-key cryptosystems. Motivated by Blom's work, in 1993, Blundo et al. [7] further investigated the polynomial-based schemes and utilized a bivariate polynomial in designing a secure key distribution for

dynamic conferences. Later, polynomial-based schemes started to attract attention from the academic community as a tool that provides a key distribution solution for symmetric encryption schemes [26], [27], [28].

2.3.1 Polynomial-based Scheme Model

A KDC creates a polynomial, $f(x, y, z, \ldots, n)$, with specific characteristics (i.e., degree, coefficients domain, modulus size, number of unknowns, etc.). The characteristics of the polynomial's structure depends on the specific system model that is demonstrated for each application domain in the following chapters. Each entity in the network is assigned a unique identity number, ID. The KDC evaluates the polynomial at each entity's ID to generate a unique keying material for every entity. Then, the KDC distributes the keying materials securely to each entity in the network.

2.3.2 Polynomial Types and Characteristics

To construct a polynomial for a specific system, we need to consider the system model and the communication flows to design the appropriate polynomial for the key distribution/generation scheme. The polynomial structures are classified based on the number of unknowns (i.e., variables) into: Univariate, Bivariate, and Multivariate polynomials.

2.3.2.1 Assumptions

- A KDC chooses a prime modulus, p, in which all polynomial coefficients are drawn from, Galois field $GF(p)$.

- The modulus p is of the size of a secret key (e.g., 100-bit or 128-bits as in AES).
- The degree of the polynomial is, t, called the threshold, which determines the security strength of the polynomial-based schemes.
- A symmetric approach was utilized in which coefficients are symmetric, $a_{i,j} = a_{j,i}$, for each entity with IDs i and j, respectively.

2.3.2.2 Univariate Polynomial

A KDC generates a polynomial in one unknown variable as: $f(x) = a_0 + a_1x^1 + \cdots + a_nx^n \bmod p$. For each entity in the network, the KDC evaluates the polynomial at entity's ID to generate a keying material, $f(ID) = k$, for the entity. The keying material generated from the univariate polynomial is a number that represents the secret key.

2.3.2.3 Bivariate Polynomial

A KDC creates a polynomial in two unknown variables, x and y, in the form: $f(x, y) = a_{0,0} + a_{1,0}x + a_{0,1}y + a_{2,0}x^2 + a_{1,1}xy + a_{0,2}y^2 + \cdots + a_{t-1,0}x^{t-1} + a_{t-2,1}x^{t-2}y + \cdots + a_{t-1,t-1}x^{t-1}y^{t-1} \bmod p$. Then, the KDC evaluates the polynomial at entity's ID in one of unknown, either x or y. the KDC generates a keying material which is a univariate polynomial, $f(ID, y) = a_0 + a_1y^1 + \cdots + a_ny^n \bmod p$, or, $f(x, ID) = a_0 + a_1x^1 + \cdots + a_nx^n \bmod p$ for the entity with identity ID. The keying material generated from the bivariate polynomial is a univariate polynomial in which the coefficients are stored in the entity's memory.

2.3.2.4 Multivariate Polynomial

A KDC creates a polynomial in n-unknown variables, as: $f(x_1, x_2, x_3, \ldots, x_n) \bmod p$. Then, the KDC evaluates the polynomial at entity's ID in one of the unknown variables and that results in a $(n-1)$-multivariate polynomial, $f(ID, x_2, x_3, \ldots, x_n) \bmod p$. Thus, the keying material generated from the multivariate polynomial is another multivariate polynomial.

From the above descriptions, it is explained that a polynomial consists of *variables, degrees, coefficients,* and a *modulus*. The number of variables determines how much information is required to generate keys. The degree of the polynomial determines the amount of computation required to generate keys as well as the number of coefficients (i.e., keying materials) to store in each entity. The modulus size determines the size of the secret keys and should be similar to the keys' sizes required to secure the communication. Thus, in designing a polynomial-based key distribution, the system model should be considered. The storage requirements of the entities are determined by the number of variables and the degree of the polynomial. The computational requirements are determined by the degree of the polynomial as well as the modules' sizes.

2.3.3 Security Properties

Polynomial-based schemes are *unconditionally* secure schemes. This means that given the attacker's unlimited resources, the attacker could not compromise the security of the polynomial-based schemes. That provides stronger theoretical security compared with the public-key schemes that are computationally secure schemes. The computational

secure schemes assume that with the current computing power, it is impossible to break a cryptosystem within a specific amount of time. This means that it would cost the attacker a lot more money to obtain a high computing power required to break a cryptosystem in less than a specific amount of time.

The polynomial-based scheme is a *threshold* scheme in which the threshold is the degree of the polynomial. If an attacker compromised a specific number of entities and obtained the secret keying materials, the attacker could use the keying materials to reconstruct the polynomial used by the KDC to generate keying materials for all entities in the network. This would compromise the security of the whole network. This is the main drawback of polynomial-based schemes which is called deterministic security. If the degree of a polynomial is t, and if an attacker compromises $t + 1$ (or more) entities and obtains the entities' keying materials, the attacker can *deterministically* recreate the original polynomial utilized by the KDC and compromise the security of the whole network. Thus, to increase the security against node-captured attacks, the degree of the polynomial needs to increase. However, increasing the degree of the polynomial also increases the storage and computational requirements of the entities. That is, there is always a tradeoff between resistance to node-captured attacks and storage/computational requirements.

CHAPTER 3

RELATED WORK

In this chapter, we discuss the state-of-art research work related to key distribution in different environments. As mentioned in Chapter 1, this dissertation, which aims at designing efficient key distribution schemes, is divided into three major sections: Wireless Sensor Networks (WSN), Fog Computing, and Blockchain Network. Thus, this related work chapter is organized into three sections as well, in which each section discusses previous works done at specific application domain.

3.1 Related Work in Wireless Sensor Networks (WSN)

The related work in the Wireless Sensor Networks (WSNs) is divided into four parts as follows: pairwise key pre-distribution schemes, hierarchical key management schemes, group key pre-distribution schemes, and key revocation schemes.

3.1.1 Pairwise Key Pre-distribution Schemes

Most key distribution and establishment schemes in WSNs create a pairwise key between two sensors. We can classify these schemes into two types: the probabilistic and the deterministic key distribution schemes. As a key distribution scheme, the probabilistic schemes do not guarantee a shred key exists between every pair of sensors in the WSNs, while deterministic schemes do. In term of resiliency against sensor-captured attack, probabilistic schemes have probabilistic security feature which means when an

attacker captured a specific number of sensors, its probabilistic that the attacker could compromise the security of WSN. However, deterministic schemes have deterministic security property which means capturing a specific number of sensors deterministically leads to compromise the security of the whole WSN.

Eschenauer and Gligor [29] proposed the first key pre-distibution scheme, more specifically the random key pre-distribution scheme. As explained in Chapter 2, the random key schemes preloaded sensors with a subset of keys, *key ring*, drawn from the key pool. The one weakness associated with the random key distribution scheme is that the secrecy of the random key pool will be compromised by an adversary if a sufficient number of key rings have been captured. Thus, many approaches have been proposed to enhance the security of the random key schemes. Chan *et al.* [30] proposed a Q-composite scheme to improve the resilience of the random key scheme. In their scheme, only in the case of two sensors sharing at least Q keys, can they establish a link-to-link communication. Even though this scheme improves the resilience against sensor capture attacks, it degrades the network connectivity since it requires at least Q shared keys to establish secure communications. There are some other random key distribution schemes to improve the resilience against sensor capture attacks. For instance, Chan *et al.* [30] proposed a pairwise key pre-distribution scheme in which each captured sensor did not reveal any information about external links. Nonetheless, their scheme is not scalable. Du *et al.* [31] proposed a random scheme assuming that the location of the sensors was available before deployment. This assumption is considered impractical for most applications. Rasheed and Mahapatra [32] proposed two key pre-distribution schemes in which bivariate polynomials were

used in generating the random key pool. However, their scheme requires the use of mobile sinks in order to ensure secure communications. In 2013, Ruj *et al.* [33] proposed a triple key establishment scheme in which any three sensors could establish triple keys among them. Recently, Yağan and Makowski [34] investigated the resiliency of WSNs against sensor capture attacks where they based their scheme on the random pairwise key distribution scheme of Chan *et al.* [30]. Ding *et al.* [35] considered prior knowledge of network characteristics and application constraints in terms of communication needs between sensor nodes and proposed methods to design key pre-distribution schemes in order to provide better security and connectivity. In 2017, Gandino *et al.* [36] proposed a q-s-composite protocol in order to exploit the best features of random pre-distribution and to improve it with lower requirements. All in all, providing high connectivity and strong resiliency against sensor capture attacks are two principal design objectives in random pre-distribution schemes. In order to provide high connectivity, the size of the key ring of each sensor needs to be large so the probability of locating overlapped keys between two sensors is high. However, favoring these features weakens the resiliency against sensor capture attacks since the adversary can recover more keys from each captured sensor.

Blom [21] proposed the first deterministic pairwise key establishment scheme using a symmetric bivariate polynomial. Furthermore, Blundo *et al.* [7] investigated the key establishment using a symmetric bivariate polynomial, $f(x,y) = a_{0,0} + a_{1,0}x + a_{0,1}y + a_{2,0}x^2 + a_{1,1}xy + a_{0,2}y^2 + \cdots + a_{t-1,0}x^{t-1} + a_{t-2,1}x^{t-2}y + \cdots + a_{t-1,t-1}x^{t-1}y^{t-1} \bmod p$, where $a_{i,j} \in GF(p)$, and $a_{i,j} = a_{j,i}, \forall i,j$. If the KDC selects a symmetric bivariate

polynomial to generate shares, $f(ID_i, y), i = 1, 2, \ldots, n$, where ID_i is the public information of each sensor, S_i, then each share, $f(ID_i, y)$, is a univariate polynomial. Since $f(x_i, x_j) = f(x_j, x_i), \forall i, j \in [0, t-1]$,a pairwise key can be shared between two sensors, S_i, and S_j. Although the deterministic key establishment scheme guarantees a shared key between any two sensors, its security is a deterministic security, also called deterministic $t - secure$, which means if the degree of the polynomial used to generate keys for sensors is $t - 1$, then capturing t or more than t sensors can compromise the security of the entire network. Increasing the degree of the polynomial can improve the security against a sensor-captured attack; however, this will increase the storage and computational requirements of the sensors. Designing a deterministic key distribution scheme with probabilistic security is the motivation of our dissertation work.

3.1.2 Hierarchical Key Management Schemes

The literature is rich with papers focusing on establishing secure key distribution schemes in WSNs. Most of the proposed schemes are implemented in a flat structure. These schemes are based on several common approaches such as random-key pre-distribution schemes [29], polynomial-based pre-distribution schemes [37], and grid-based pre-distribution schemes [38].

In large-scale networks, researchers observed that hierarchal WSNs perform better than flat networks in terms of communication overhead and scalability [39]. This is in part due to the ability of aggregating data from large numbers of sensor nodes onto relay nodes and then forwarding them to destination nodes in fewer hops [39], [40], [41], [42].

Shen *et al.* [43] proposed a key distribution scheme in hierarchal WSNs based on a symmetric bivariate polynomial. Although their scheme is scalable, secure, and lightweight, it shows high communication and computational overhead. On the other hand, Kumar *et al.* [44] proposed a symmetric/asymmetric key pre-distribution scheme based on a hardware chip, called a Trusted Platform Module (TPM) that is added to the CHs and sensors in the hierarchal WSNs. This chip-based scheme resists physical attacks such as node capture attacks, node injection attacks, and node impersonation attacks. However, the security of this scheme depends on the TPM chip in devices (Cluster Heads (CHs) and sensor nodes). The TPM chip embedded in every sensor increases the whole network cost, and that is not a practical solution to be considered in a large WSN [45]. In [46], Mahmood *et al.* proposed a polynomial subset-based multiparty key management system for limited-resource devices such as WSNs and the Internet of Things (IoT). In the polynomial generation phase, their proposed scheme uses an XOR operation instead of the expensive multiplication operations to reduce the computation overhead. Although the scheme shows a reduction in the storage and computation overhead, it has a high communication overhead and requires regenerating a new polynomial when a sensor node joins or leaves the network. In [47], the authors proposed a multivariate polynomial-based key management scheme in which a base station creates a pool of random symmetric trivariate polynomials; then each CH chooses a trivariate polynomial and generates shares for all sensors in its cluster, which are bivariate polynomials. The CH hashes the shares before sending them to the sensors to ensure their integrity. If node i and node j want to communicate, they use their shares to create the pairwise keys, $f(i, j) = f(j, i)$. Thus,

each sensor stores a bivariate polynomial's coefficients, which requires a large storage space. In [48], Bahrami *et al.* proposed a hierarchical key pre-distribution scheme in a fog network to provide secure communication between end-devices (i.e., constrained devices) in a fog cluster, and between end-devices and fog nodes (i.e., CHs), which is simulating the hierarchical WSN's schemes. Their proposed scheme, which is based on a residual design, shows less memory requirements and enhances the scalability of the network. However, their key pre-distribution scheme follows the probabilistic schemes in which a shared pairwise key between end-devices in a fog cluster is not guaranteed; so, in case there is no shared key between two end-devices, they will start a path-key discovery phase in which they need to find an intermediary node that shares a key with both of them, this way introducing a communication overhead. In [49], Albakri *et al.* proposed a hierarchical polynomial-based key management scheme in fog computing. Although their proposed scheme exhibits a good performance in terms of communication, computation, and storage space for IoT devices (i.e., constrained devices), it does not consider communication links between IoT devices. Hamsha and Nagaraja in [50] proposed a lightweight threshold key management scheme in WSNs. The proposed scheme is based on Shamir's secret key sharing scheme to generate shares for all nodes in the network. They divided the network into multiple levels, and at each level the base station is responsible for selecting a polynomial, secret keys, generating shares, and updating thresholds. In addition, all sensors are preloaded with a network key that is utilized for secure data transmission between sensor nodes. Although the scheme seems lightweight in terms of storage, it is vulnerable to sensor capture attacks. Capturing one sensor enables an attacker to obtain

the network key and that leads to compromising the security of the WSN. In [51], Kumar *et al.* proposed a key pre-distribution scheme in WSNs based on combinatorial design. Their scheme improves the overall network resiliency and ensures network connectivity in case cluster heads or sensor nodes are compromised by an attacker. Also, they adopted a symmetric design to reduce the storage requirements of cluster heads. On the other hand, their proposed scheme exhibits high communication and computation overhead at the shared key discover phase. In our proposed scheme demonstrated in Chapter 4, we preloaded all network nodes (i.e., cluster heads, sensors) with tokens that enable each node to establish shared keys independently without requiring additional information to be transferred to establish the keys.

3.1.3 Group Key Pre-Distribution Schemes (GKPS)

Blundo *et al.* [7] proposed a non-interactive k-secure m-conference protocol based on a multivariate polynomial, $f(x_1, x_2, \ldots, x_m)$. Because each share, $f(ID_i, x_2, \ldots, x_m)$, is a polynomial involving $m - 1$ variables with degree k, each sensor needs to store $(k + 1)m - 1$ coefficients. The storage space of each sensor is exponentially proportional to the size of the conference that deems this protocol impractical. Khan *et al.* [52] proposed a pre-distribution scheme using a symmetric matrix and a generator matrix of maximum rank distance to establish pairwise keys for sensor nodes. Sheu and Cheng [53] proposed a hop by hop authentication scheme for path key establishment in WSN that enabled sensor nodes to identify malicious nodes and detected false data that were injected in the network. Recently, Harn and Gong [54] and Harn and Hsu [55] proposed

group key establishment schemes using a special type of multivariate polynomials. The advantage in using this special type of polynomial for group key establishment is that the storage requirement of each sensor is fixed and is independent of the size of the WSNs. As we mentioned earlier, there is one problem associated with all polynomial-based key distribution schemes which is the security of these schemes is deterministic.

3.1.4 Key Revocation Schemes

As sensor nodes are deployed in hostile and unattended environments, they are exposed to various attacks. To secure WSNs and ensure the confidentiality of data transmitted in the network, it is crucial to implement a revocation mechanism to exclude the compromised nodes from participating in network activities or revealing the content of secure messages. In [56], Ge *et al.* classified the revocation schemes into two categories: centralized and distributed schemes. In the distributed revocation schemes, sensor nodes collaborate with each other to exclude compromised nodes using voting techniques. On the other hand, the centralized approach transfers the revocation process into a central authority (i.e., base station) that becomes responsible for detecting the compromised nodes and removing compromised keys in the sensor nodes. Although the centralized approach exhibits a single point of failure, since all revocations are handled by the central authority, it shows a better performance (i.e., storage, communication, and computation overhead) than distributed revocation mechanisms [56], [57]. In our proposed scheme, we adopt the centralized approach for key revocation. However, the detection mechanism is not in the scope of this dissertation work. We assume that the cluster head (CH), which monitors the

node activities, can identify the misbehaving node and is able to revoke the compromised node from the network.

3.2 Related Work in Fog Computing

Fog computing has a hierarchical structure [58], in which the top layer is the cloud with its data centers and servers, the middle layer is Fog Nodes (FN) which include the base station, servers and routers that handle the computing, networking and storage requests, the lowest layer includes End-Users (EU) nodes which could be any IoT devices (e.g. sensors, smart phones or vehicles), objects or infrastructure such as buildings or homes. Security issues in fog computing have not been received too much attention. There is no significant research on key management in fog environment.

In [59], Lee *et al.* described security and privacy issues that need to be considered in fog computing and mentioned possible attacks in such an environment with possible countermeasures; however, they do not include a cryptographic model to provide such security services. Lu *et al.* [60] introduced a lightweight data aggregation scheme which can resist to data injection attack, but they do not consider the key distribution mechanisms required to provide such security services. Amor *et al.* [61] proposed a privacy-preserving authentication scheme in Fog environment based on a public key cryptosystem. However, public-key cryptosystem has expensive computation which is considered impractical to be implemented in fog end-user devices due to end-user devises' inherent characteristics (i.e. limited memory, processing and battery power). Alrawais *et al.* [62] proposed a secure communication scheme in Fog environment based on ciphertext-policy

attribute-based encryption scheme, which is a public-key cryptosystem that uses digital signature techniques. The scheme is designed to provide authentic and confidential communication between fog nodes. However, they do not consider the communication between fog nodes and user-nodes. Porambage *et al.* [63] proposed a proxy-based authentication and key establishment protocol in IoT. In [63], the heavy cryptographic operations needed to establish the end-to-end secure connection are delegated to sensor nodes' nearest neighboring devices (i.e. proxies) that has more capabilities (i.e. storage, computation and battery power) than sensor nodes (i.e. IoT devices) . The scheme in [63] is based on Diffie-Hellman (DH) protocol [64] and Shamir secret sharing scheme [65] which is a (n, k) threshold scheme in which n proxies process a polynomial share, and k polynomial shares are enough to reconstruct the DH keys. The protocol in [63] shows high communication, computation and storage requirements for sensors.

3.3 Related Work in Blockchain Network

Recently, blockchain technology has attracted tremendous interest from both academia and industry. When it comes to key management, most of the blockchain implementations are based on the Public-Key Infrastructure (PKI). In this section, we review various key management schemes that utilized blockchain technology.

In [15], Ao *et al.* proposed a key management scheme in a vehicular communication system based on blockchain technology. The communication in the vehicular network is based on broadcast communication that utilizes a group key to provide a secure communication between vehicles within a specific geographical area (i.e., security

domain). Since vehicles depart frequently from one security domain to another, group keys must be updated. This update process is managed by a security manager which communicates with a central authority to validate certificates and generate new cryptographic materials for new joining vehicles. This process introduces a delay in key transmission between two security domains. Thus, Ao *et al.* adopted blockchain as a way to transmit keys between security domains. However, they follow the public blockchain, *i.e., bitcoin*, which requires extensive operations for the mining process, and that increases the processing overhead on security managers and delays to transfer keys. Our polynomial-based scheme (in Chapter 6) eliminates PKI overhead by preloaded entities with tokens that facilitates the establishment of shared keys and transmission of data securely within the network. As a result, this reduces the time to process transactions and transfer data. Lin *et al.* [66] introduced an ID- based linearly homomorphic signature scheme in which an ID-based cryptosystem simplifies the key management processes compared to a certificate-based public-key infrastructure used to authenticate data stored in the blockchain. On the other hand, the linearly homomorphic signature scheme is utilized to do computations over authenticated data. Their proposed scheme allows users not only to authenticate data stored in the blockchain but also to verify the correctness of the results of the operations performed on the authenticated data. Also, their scheme eliminates the certificate-based cryptosystem because it requires the frequent validation of certificates and that introduces overhead in the system. In a similar way, our key management scheme eliminates PKI due to certificate processing overhead by relying on a pre-distribution scheme to provide data confidentiality in a permissioned blockchain. In [67], Dorri *et al.* integrated blockchain

technologies with Internet-of-Things (IoT) in a smart home environment. They proposed a scheme in which blockchain is utilized to provide data confidentiality and integrity to both the owner of the home devices and a third party, which needs to process the data stored in the blockchain. They provide a lightweight and secure blockchain architecture for IoT devices in which security and privacy services for IoT devices eliminates the overhead of the blockchain. However, the integration of blockchain and IoT needs to consider IoT's characteristics such as limited memory and processing power. Since most of the blockchain technologies are based on PKI, it requires more resources to send and validate transactions. Our proposed scheme can be applied to the IoT environment since it is a lightweight scheme in terms of storage and computation requirements.

CHAPTER 4

WIRELESS SENSOR NETWORKS (WSNS)

Wireless sensor networks (WSNs) have been deployed in numerous settings, such as in health/traffic monitoring [68], the military domain [69], and in hazardous environments for data acquisition purpose [70]. Due to the sensitivity of the data that are transferred in WSNs, these data need to be protected; otherwise, adversaries can easily capture the data and recover the sensitive information that is being exchanged among sensors. In order to fulfill security services such as data encryption and data authentication, the source and destination nodes must share a secret key prior to transmitting any data over the WSNs. Establishing secret keys among sensors is called the key distribution/establishment in WSNs. We assume that each sensor is randomly deployed into a geographical area so that their relative locations cannot be pre-determined. Furthermore, we acknowledge that sensors are limited-resource devices, which means they have limited memory space, computational power, and battery life. As a result, the design of a key distribution scheme in WSNs must take these limitations into account. Thus, when designing key distribution schemes in WSNs, there are some objectives that need to be satisfied, such as low memory requirements, low computational and communication overhead, and high connectivity and robustness against node capture attacks. In this chapter, we propose the first polynomial-based key distribution schemes with probabilistic security. Our proposed solutions designed for a Hierarchical Key Management Scheme and

Non-Interactive Group Key Pre-Distribution Scheme (GKPS) for End-to-End Routing in WSNs.

Motivation

Data transmitted in the WSN need to be protected; otherwise the data collected in the network are also available to attackers. To secure data transmitted between sensors in the WSN, sensors must share keys with other sensors before transmitting any data. These keys are utilized to protect the data from attackers. Polynomial-based schemes have been adopted to establish and distribute keys to sensors in WSNs. The motivation of our work is based on two reasons:

First, our approach of using a polynomial-based key distribution scheme to generate tokens (i.e., keying materials) for sensors is to simplify key establishment tasks in wireless sensor communication. Since sensors are constrained devices, we aim at reducing the amount of information that needs to be transmitted and stored by sensors as well as reducing the computational processing overhead. Tokens preloaded into sensors facilitate establishing pairwise shared keys between any two sensors non-interactively. Consequently, the communication overhead of our scheme must be much shorter than most nonpolynomial-based designs. This is because nonpolynomial-based schemes need to exchange information interactively in order to establish pairwise keys that result in high communication overhead. More details are discussed in the performance analysis section.

Second, our work is motivated by the fact that all polynomial-based pre-distribution schemes are inherently deterministic security schemes that are vulnerable to sensor- captured attacks. Thus, if an attacker captures a specific number of sensors, the attacker can

reconstruct the polynomial used to generate keying materials (i.e., tokens) in the network and that leads to compromising the security of the whole network. In this work, we aim at designing a polynomial- based scheme that resists sensor-captured attacks. We propose a polynomial-based scheme with probabilistic security that reduces sensor capture attacks. More specifically, capturing a specific number of sensors has a very low probability for an attacker to reconstruct the polynomial and compromise the WSNs. We can adjust the system's parameters to lower this probability and enhance the security of the network.

4.1 Part1: Hierarchical Key Management Scheme with Probabilistic Security in a Wireless Sensor Network (WSN)

A WSN consists of sensor nodes that are usually deployed in unattended environments to sense events or specific phenomena and relay such data to other sensors. WSNs can be classified into two types: *flat* and *hierarchical*. In flat WSNs, all sensors have the same capabilities to collect data and forward them to other sensors in the network. In hierarchical WSNs, devices are organized into a hierarchy based on their capabilities: sensor nodes with their limited capabilities are located in the bottom of the hierarchy; cluster heads (CHs) are located in the middle of the hierarchy and have more capabilities than those of sensor nodes; and a mobile sink (or a base station) has the largest capabilities and is located at the top of the hierarchy. In the hierarchical structure, sensor nodes are responsible for forwarding data to the CHs, which process the data and send them to the base station, where further analysis on the collected data can take place [43].

Since sensor nodes are limited in their memory, as well their processing and

battery power, adding security services is a challenging task. Incorporating key distribution protocols in WSNs must accommodate the sensors' physical limitations. Utilizing asymmetric cryptographic schemes [71] , [72], [73] is considered impractical as they require extensive computation and large storage that are not suitable for implementation in sensors due to their inherit characteristics (*i.e.*, limited memory, processing, and battery power). There are several approaches to designing secure key distribution schemes in WSNs. One approach is to preload all sensors with one master key. This approach provides high network connectivity, low storage requirements, and no communication/computation overhead. However, the network becomes vulnerable to node capture attacks as capturing one sensor compromises the security of the entire network. Another approach is to preload each sensor with a pairwise key that is shared between two sensors. In this approach, each sensor needs to share a pairwise key with every other sensor in the network. This approach can resist node capture attack, but the storage requirement is linearly proportional to the network size. Thus, this approach is impractical to be implemented in a large network.

In this work, we propose a polynomial-based key distribution scheme in a hierarchical WSN with probabilistic security. Our proposed scheme follows the *hierarchical key management structure* in which sensors in a WSN are classified into multiple clusters and keys are generated based on the hierarchical structure. In summary, the proposed scheme has the following features [74]:

- It is the first polynomial-based key distribution scheme with probabilistic security.
- It guarantees a shared key between two sensors.

- If the degree of the chosen polynomial is $t - 1$, then capturing t or more than t sensors has a very low probability of compromising the security of the WSN.
- It provides keys to support three types of communication: sensor-to-sensor communication, sensor-to- cluster head (CH) communication, and CH-to-sink communication.
- It provides a revocation mechanism to ensure the confidentiality of data transferred in the network.
- It has a hierarchical key management structure, which means it minimizes the storage requirements of each sensor, CH, and the sink.

4.1.1 Network Model

In a hierarchal structure, the WSN is partitioned into several clusters, depending on the network's application. In each cluster, there is a CH that serves all sensor nodes in its cluster. To transmit data, each sensor node sends data to its local CH. Next, the CH processes the data and aggregates it, then forwards it to the base station. The hierarchal model with different types of sensors has been utilized in various environments. For example, in the military environment, different types of sensors are utilized to collect data (e.g., image, sound, motions) for different purposes such as intrusion detection, chemical and biological threat detection, and object presence detection [75], [76]. The sensors can also be classified based on their physical properties such as motion sensors, thermal sensors, pressure sensors, chemical sensors, etc. [77]. The model of our proposed hierarchical key management scheme is shown in Figure 9.

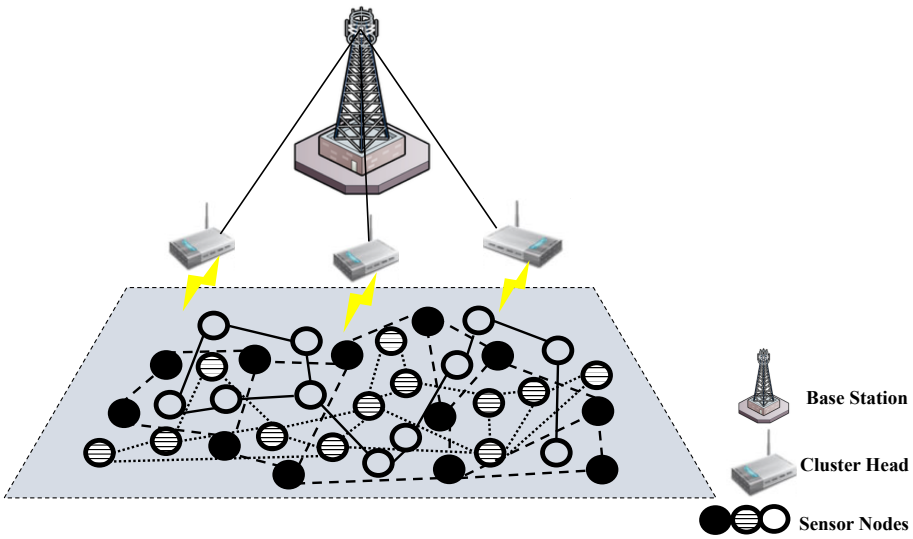

Figure 9: The network model of the proposed hierarchal key management scheme.

4.1.2 Model of Proposed Scheme

Instead of adopting a flat key management [29], [37], [38], [40] – [42], our proposed scheme uses the hierarchical key management model [78], [79]. In such models, sensors are distributed into different clusters. Each cluster has a cluster head (CH). All CHs are connected to a sink. Collected information by each sensor node can be transmitted to its neighbor sensor node and to its CH. Finally, all collected data are sent to the sink by the CH. Arranging data in a WSN in such a hierarchical structure has several advantages [80]. First, in a hierarchical network, the CHs and the sink manage most communication traffic of the network. Sensors are woken up only when they are needed for data transmission or data collection. This can reduce energy consumption. Furthermore, the CH is able to conclude the local information since all collected information

passes through it. Finally, the CH transmits most data, so more communication channels can host more sensors in the network. Consequently, the hierarchical WSN has better scalability and is more efficient than the flat WSN.

In our proposed hierarchical key management structure, tokens (*e.g.*, keying materials) are generated from top to the bottom. The Key Generation Center (KGC), or Key Distribution Center (KDC), first selects a trivariate polynomial, $F(x, y, z)$, and uses it to generate all tokens in the WSN. The token of the sink is the trivariate polynomial. Then, the KGC uses the trivariate polynomial to generate tokens, which are bivariate polynomials for all cluster heads. Similarly, the KGC uses the bivariate polynomial of each CH to generate tokens, which are univariate polynomials for all sensor nodes in the cluster. In our proposed structure, each upper level device in a WSN is able to access tokens of the lower level devices. For example, the sink knows the tokens of all CHs and each CH knows all the tokens of its sensor nodes.

In data aggregation, a sensor node aggregates the reported values from its children and forwards the aggregated value to its parent. The hierarchical key management of our proposed scheme can provide keys to support three types of secret communications in a WSN:

(1) *unicast communication* in which a node sends data to a single node or its cluster head.

(2) *local broadcast* in which the cluster head sends data to all the nodes in the cluster.

(3) *global broadcast* in which the sink sends data to all the nodes in the network.

4.1.3 Proposed Scheme

We proposed a polynomial-based key distribution scheme with probabilistic security. The three main steps in our pre-distributed key management scheme are the token generation, key establishment, and key revocation. The following subsections explain each step in detail.

4.1.3.1 Token Generation

A Key Generation Center (KGC) initially selects a prime modulus, p, and a trivariate polynomial, $F(x, y, z)$, to generate sensors' tokens in a WSN, where p has the same size of keys needed in secret communication. We assume that the degree of x is $k-1$, the degree of y is $t-1$, and the degree of z is $h-1$, where these parameters (*i.e.*, $k-1, t-1$ and $h-1$) are the *thresholds* of the polynomials used to generate tokens. In addition, these parameters determine the strength to *resist* the sensor capture attack. The trivariate polynomial is retained by the sink of the network. We assume each device in the WSN (*e.g.*, the sink, CH and, sensors) has a unique ID.

Each CH with a cluster identity, ID_C, has a token that is a bivariate polynomial, $F(ID_C, y, z) \bmod p$. Each unique bivariate polynomial is kept by each CH. Note that the properties of this type of asymmetric bivariate polynomial can be found in [81]. Moreover, tokens of sensor nodes in the same cluster are generated by a bivariate polynomial. For example, the token of a sensor node with the identity, $id_j \in ID_C$, is $F(ID_C, id_j, z) \bmod p$ and $F(ID_C, y, id_j) \bmod p$, which are two univariate polynomials. Each sensor node needs to store the coefficients of two univariate polynomials.

4.1.3.2 Key Establishment

Our proposed scheme provides two types of keys: *unicast* and *broadcast* communication keys. The unicast keys are used to support sensor-to-sensor, sensor-to-CH and CH-to-Sink communications. The broadcast keys are either *local keys* used by the CHs to send messages to the sensors in their cluster, or *global keys* used by the sink to broadcast a message to all sensors in the network. The following subsections demonstrate the key establishment process.

(a) Unicast Communication Keys

- *Key between two sensor nodes*. According to [81], any two sensor nodes in the same cluster can share a pairwise key. For example, two sensor nodes with the following identities and tokens, $id_j \in ID_C, F(ID_C, y, id_j) \bmod p, F(ID_C, id_j, z) \bmod p$, and $id_k \in ID_C, F(ID_C, y, id_k) \bmod p, F(ID_C, id_k, z) \bmod p$, respectively, can share a key $F(ID_C, id_k, id_j)] \bmod p$ if $id_j > id_k$ or $F(ID_C, id_j, id_k) \bmod p$ if $id_j < id_k$, as seen in Figure 10 . Thus, any node can send data secretly to any other node in the same cluster.

- *Key between a sensor node and its CH*. Any CH can use its bivariate polynomial to share a pairwise key with any sensor node in its cluster. For example, the CH with cluster identity, ID_C, and its bivariate polynomial, $F(ID_C, y, z) \bmod p$, can share the key, $K_{C,id_j} = F(ID_C, ID_C, id_j) \bmod p$, with a sensor node with the consequent identity and tokens, $id_j \in ID_C, F(ID_C, y, id_j) \bmod p, F(ID_C, id_j, z) \bmod p$.

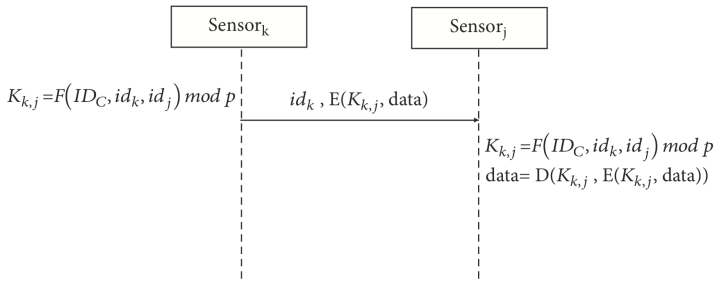

Figure 10: Key establishment between sensor with idk and sensor with $id_j (id_k < id_j)$.

- *Key between a sink and each CH*. The sink can use its trivariate polynomial to share a pairwise key with any CH in a WSN. For example, the sink with the trivariate polynomial, $F(x, y, z) \bmod p$, can share the key, $K_{S,C} = F(ID_C, ID_C, ID_C) \bmod p$, with a CH with identity, ID_C, and token, $F(ID_C, y, z) \bmod p$.

(b) Local Broadcast Key

A local broadcast key, K_{LB}, can be determined and sent to each sensor node separately as $E_{K_{C,id_j}}(K_{LB})$ by their CH; the broadcast key, K_{LB}, is encrypted under the shared key, K_{C,id_j} , with each sensor node.

(c) Global Broadcast Key

A global broadcast key, K_{GB} , can be determined and sent to each cluster head separately as $E_{K_{S,C}}(K_{GB})$ by the sink. Thus, the broadcast key, K_{GB}, is encrypted under the shared key, $K_{S,C}$ with each cluster head. After receiving the global broadcast key, each CH forwards it to each sensor node separately as $E_{K_{C,id_j}}(K_{GB})$, which means that the global broadcast key, K_{GB} is encrypted under the shared key, K_{C,id_j}, with each sensor

node.

4.1.3.3 Key Revocation

If an attacker compromises a sensor node, he can get all of the stored keys including the unicast and broadcast keys. Thus, it is important to utilize a key revocation scheme. The sink creates a *Node Revocation List (NRL)* that includes the IDs of all revoked nodes. The NRL is initially empty and is populated whenever a compromised node gets detected. This list is stored on each device in the WSN. The NRL is checked for any messages exchanged in the network to ensure that all current members of the network are valid/non-compromised nodes. We adopt the node revocation list from the scheme of Wang *et al.* [82] since it is an efficient and simple way to allow nodes (*i.e.*, the sink, CH, sensors) to identify the compromised nodes and exclude them from the network [56], [57].

Our revocation mechanism works as follows. The CH are responsible for monitoring the sensors' activities and detecting misbehaved sensors. If a malicious sensor is detected, the CH will add that sensor's ID into its NRL. Since the broadcast keys stored in the compromised sensor are revealed by the attacker, the CH must update the local broadcast key, K_{LB}, and send it to a non-compromised sensor (not in the NRL) in its cluster separately and encrypted by the shared pairwise key between a sensor and its CH as,$E_{K_{C,id_j}}(K_{LB})$. Then, the CH must send the updated NRL encrypted by the local broadcast key, $E_{K_{LB}}(NRL)$, to all sensors in the cluster. So, each non-compromised sensor can decrypt the message using the local broadcast key, which authenticates the CH, and then updates its NRL. After that, the CH sends a request to the sink to update the

global broadest key, K_{GB}. It also sends the updated NRL to the sink, encrypted by the pairwise key as, $E_{K_{S,C}}(NRL)$, where , $K_{S,C}$, is the pairwise key between the CH and the sink. The sink will authenticate the message sent by the CH, update its NRL, create a new global broadcast key, K_{GB}, and send it to each CH in the network encrypted by the pairwise key between CH and the sink as, $E_{K_{S,C}}(K_{GB})$. Once each CH receives the message from the sink, it authenticates the message using the pairwise key, $K_{S,C}$, updates the global broadcast key , K_{GB}, and sends a local broadcast message to all nodes in its cluster to update the global broadcast key as $E_{K_{LB}}(K_{GB})$. This way, only sensor nodes that have the updated local broadcast key can decrypt the message and store the updated global broadcast key.

Our proposed key establishment scheme is a polynomial-based scheme. Thus, we do not need to adopt a sophisticated revocation mechanism since nodes do not store pairwise keys. Instead, they store the polynomial's coefficients that are used along with the node ID to create the pairwise key. Before that, a sensor's ID is checked to make sure that it is not listed in the NRL, otherwise, this communication is terminated. Threshold is the intrinsic part of the polynomial-based scheme, so if the number of compromised nodes approaches the threshold value, new tokens must be generated.

4.1.4 Security Analysis

In our hierarchical key management structure, there are three types of devices in a WSN. These are, the sensor nodes, cluster heads (CHs) and a sink. Each WSN hosts a large number of sensor nodes, which are located at the lowest level of the structure. Each

sensor node has stored unique secret tokens that can be used to support secure communications with the CH or any other sensor node in the cluster. In every network, there exists multiple CHs where each CH manages communications occurring within the cluster. Each cluster has stored a unique secret token that can be used to support secure communications between any sensor node in the cluster and the sink. There is only one sink (i.e., also called *"Base Station"*). The sink is located at the highest level of the hierarchical structure and holds a unique token that can be used to support secure communications with any cluster head.

4.1.4.1 The Attack Model

The attack model of the proposed scheme is divided into two categories: sensor capture attacks and common network attacks.

(a) Sensor Capture Attacks

Due to the vulnerable and open environment where sensor nodes are deployed, it becomes easy to physically capture the sensors. In addition, sensor nodes are not equipped with tamper-proof hardware, and that enables attackers to obtain tokens stored in the captured sensors, which leads to serious security issues. Thus, we aim at decreasing these attacks as much as we can.

if an attacker compromises the token of the sink, the security of our proposed scheme breaks completely. Thus, the sink needs to be well protected. Since there is only one sink, we can adopt a sophisticated mechanism, such as a hardware-based tamper-proof technology to strengthen its security.

If an attacker compromises the token of each CH, all tokens of sensor nodes within the cluster will be compromised. However, the tokens of sensor nodes located in other clusters will be intact. The following theorem discusses the security if the attack compromises multiple CH tokens.

Theorem 1. *If the attacker captures k CH tokens, the attacker can recover the trivariate polynomial used to generate all secret tokens.*

Proof. The trivariate polynomial used to generate all tokens is $F(x, y, z)$ where the degree of x is $k-1$, the degree of y is $t-1$, and the degree of z is $h-1$. The token of each cluster head with cluster identity, ID_C, is a bivariate polynomial, $F(ID_C, y, z) \bmod p$. Assume that k tokens of CHs with their identities, $ID_{C,i}, i = 1, 2, .., k$, have been compromised by an attack. Then, following the Lagrange interpolation formula, the attacker can obtain $\sum_{i=1}^{k} F(ID_{C,i}, y, z) \prod_{j=1, j\neq i}^{k} \frac{(x-ID_{C,j})}{(ID_{C,i}-ID_{C,j})} \bmod p = F(x, y, z)$. However, fewer than k tokens of CHs cannot obtain the trivariate polynomial.

Note. If we limit the number of CHs to be fewer than k, the above attack can never occur.

In this following discussion, we divide sensor capture attacks into two types, the situation when (a) all capturing sensors belong to the same cluster and when (b) not all capturing sensors belong to the same cluster.

Theorem 2. (Sensor Capture Attack I) *If the attacker has captured t sensor nodes belonging to the same cluster, it can recover the bivariate polynomial used to generate tokens of sensor nodes in the cluster.*

Proof. The polynomial used to generate tokens of sensor nodes belonging to the same cluster is a bivariate polynomial, $F(ID_C, y, z)$ with degree $t-1$ in y and $h-1$ in z.

Knowing t sensors' token values, $F(ID_C, id_j, z), j = 1, 2, \ldots, t$, from the Lagrange interpolating formula, the attacker can recover the bivariate polynomial used to generate tokens in this cluster as $\sum_{j=1}^{t} F(ID_C, id_j, z) \prod_{i=1, i\neq j}^{t} (\frac{(y-id_i)}{(id_j-id_i)}) \bmod p = F(ID_C, y, z)$. However, acquiring less than t tokens does not grant the recovery of the bivariate polynomial.

Note. This sensor capture attack can only be applied if all captured sensor nodes are in the same cluster. This condition decreases the possibility of a sensor capture attack occurring since captured sensor nodes randomly belong to different clusters in WSNs. In summary, our proposed scheme effectively reduces the risk of a sensor capture attack since this attack only works if two conditions are satisfied simultaneously, (a) having captured t or more sensor nodes; and (b) having at least t sensor nodes belonging to the same cluster in all captured nodes. Furthermore, if we limit the number of sensor nodes in each cluster to be less than t, then this attack can never occur.

Theorem 3. (Sensor Capture Attack II) *If the attacker has captured m sensors (i.e., $m > tk$) among which at most t sensors belong to the same cluster, he can recover the trivariate polynomial used to generate sensor tokens.*

Proof. The trivariate polynomial used to generate all tokens is $F(x, y, z)$, where the degree of x is $k - 1$, the degree of y is $t - 1$, and the degree of z is $h - 1$. Recall that the polynomial used to generate the tokens of sensors is a bivariate polynomial, $F(ID_C, y, z)$ with degree $t - 1$ in y and $h - 1$ in z. According to [81], from each captured sensor with tokens, $F(ID_C, id_j, z) \bmod p$ and $F(ID_C, y, id_j) \bmod p$, we can establish at most $t + h$ linearly independent equations in terms of the coefficients of the trivariate polynomial,

$F(x, y, z)$. Thus, there are at most $t(t + h)$ linearly independent equations that can be established from the t captured sensors belonging to the same cluster. We assume that there are m captured sensors among which (at most) t sensors belonging to the same cluster exist. If the number of coefficients of the trivariate polynomial, $F(x, y, z)$, is larger than the number of equations available to the attacker that is, $thk > m(t + h)$. Then, these m captured sensors cannot recover $F(x, y, z)$. On the other hand, if $m > thk/(t + h)$, it can solve the trivariate polynomial used to generate the tokens of the sensors. Furthermore, from [81], since $t(t + h) > th$, we have $m > tk$. In other words, this attack needs to capture far more sensors than the previous attack to compromise the bivariate polynomial used to generate tokens for each class.

Note. This sensor capture attack is much harder than the previous attack since (a) it needs to capture far more sensors than the previous one, and (b) among these captured sensors, there are at most t sensors belonging to the same cluster.

(b) Common Network Attacks

This section describes several common network attacks we consider when designing our proposed scheme.

(1) Impersonation Attack. It is an attack in which the adversary assumes the identity of a legitimate entity in the wireless sensor network. The proposed scheme allows sensors to exchange their identities over the network to establish the pairwise keys independently. If an attacker tries to send a fake identity and pretend to be a legitimate sensor, the attacker will not obtain any information from sensors nodes because the data adversary sent does not come from the same mathematical structure (*i.e.,* the polynomial used by KGC

to generate tokens). As a result, its message will be dropped and legitimate sensors will terminate the communication. Because the attacker does not possess any valid token, the attacker will never be able to establish a pairwise key and send valid data that sensors can decrypt with their pairwise keys.

(2) Replay Attack. It is an attacker who captures a message and tries to replay it to sender to confuse sender and obtain information. The proposed scheme preloaded sensors with tokens that enable sensors to compute pairwise keys independently. No additional information is required to establish the keys other than the sensors' identities. Utilizing nonce for sensors communications eliminates the replay attack [83].

(3) Key Exposure Attack. Exchanging keys over the network could lead to key exposer attacks [83]. Since our proposed scheme is predistribution scheme in which sensors are preloaded with tokens before deployed into the WSNs, there are no keys transferred over the network. Our aim is to reduce the amount of information that need to be exchanged between sensors to establish the keys. Thus, our proposed scheme resists such attacks.

4.1.5 Performance

In this section, we evaluate the performance of the proposed scheme in terms of storage, computation, and communication overhead. In addition, the probabilistic property of the proposed scheme is explained.

4.1.5.1 Storage requirement

In our proposed scheme, only the sink needs to store a trivariate polynomial, $F(x, y, z)$, where the degree of x is $k-1$, the degree of y is $t-1$, and the degree of z is $h-$

1. In other words, the storage of the sink is kth coefficients in $GF(p)$. Each CH with cluster identity, ID_C, needs to store a bivariate polynomial, $F(ID_C, y, z) \bmod p$. The storage requirement of each cluster head is th coefficients in $GF(p)$. Each sensor node with identity, $id_j \in ID_C$, needs to store two univariate polynomials, $F(ID_C, y, id_j) \bmod p$, and $F(ID_C, id_j, z) \bmod p$. The storage requirement of each sensor node is $t + h$ coefficients in $GF(p)$. In summary, in our proposed hierarchical key management scheme, each sensor node located at the lowest level only needs to store a minimal number of coefficients, but the sink located at the highest level needs to store the most coefficients.

4.1.5.2 Computational requirement

In the following discussion, we evaluate the computational requirements of various communication keys. Horner's rule [84] can be used to reduce the computational cost in the polynomial evaluation. According to Horner's rule, evaluating a univariate polynomial of degree $h - 1$ needs $h - 1$ multiplications and h additions.

- *Key between two sensor nodes* - Two sensor nodes, with their identities, id_j and id_k in the same cluster can share a pairwise key, $F(ID_C, id_k, id_j)$ if $id_j > id_k$ or $F(ID_C, id_j, id_k)$ if $id_j < id_k$. For example, if $id_j > id_k$, sensor node with identity, id_j, can use its token, $F(ID_C, y, id_j)$, which is a univariate polynomial in y having degree $t - 1$ to obtain the shared key, $F(ID_C, id_k, id_j)$. It needs $t - 1$ multiplications and t additions. Similarly, sensor node with identity, id_k, can use its token, $F(ID_C, id_k, z)$, which is a univariate polynomial in z having degree $h - 1$, to obtain the shared key. It needs $h - 1$ multiplications and h additions.

- *Key between sensor node and cluster head-* Any cluster head with identity, ID_C, can use its bivariate polynomial, $F(ID_C, y, z)$, to share a pairwise key, $K_{C,id_j} = F(ID_C, ID_C, id_j)$ with any sensor node with identity, id_j, in the same cluster. The bivariate polynomial, $F(ID_C, y, z)$, has $t-1$ degree in y and $h-1$ degree in z. The cluster needs th multiplications and $th+t-1$ additions.

- *Key between sink and each cluster head* - The sink can use its trivariate polynomial, $F(x, y, z)$ to share a pairwise key, $K_{S,C} = F(ID_C, ID_C, ID_C)$ with any cluster head with identity, ID_C. The trivariate polynomial, $F(x, y, z)$ has $k-1$ degree in x, $t-1$ degree in y and $h-1$ degree in z. The sink needs $k(th+1)$ multiplications and $kth+kt-1$ additions.

4.1.5.3 Communication overhead

The proposed scheme has a low communication overhead for key establishment. After deployment, no information needs to be transmitted to establish the shared pairwise keys except the sensors' IDs and that is a crucial step for self-organization protocols in WSN. Thus, there is no such overhead in key distribution schemes [43]. On the other hand, updating broadcast keys, which involves sending new broadcast keys to each sensor using the pairwise keys, may introduce some communication overhead.

4.1.5.4 Probabilistic security

Our proposed scheme is the first polynomial-based key distribution scheme with probabilistic security. Unlike all polynomial-based schemes, in which capturing t or more

than t sensors can recover the polynomial of degree $t-1$, which led to compromising the whole network security, the random deployment of sensors loaded with different polynomial structures make it difficult to guarantee that t captured sensors belong to the same class. The proposed scheme allows sensors from the same cluster to communicate with each other since their tokens/shares are generated from the same polynomial structure. Thus, sensors from different categories can not communicate with each other because they are pre-loaded with shares that are created from different polynomials. In order to reveal the polynomial used to generate the shares for sensors, the attacker needs to collect at least t sensors that all belong to the same cluster. In that way, the probability of finding such sensors is very low, increasing the difficulty of sensor-capture attacks. This increased difficulty leads to the enhancement of the security of the WSN. Below we show the statistical analysis of the probabilistic security of the proposed scheme.

In a WSN, if n is the number of sensor nodes and l is the number of clusters, then, r is the number of sensor nodes in each cluster (*i.e.*, $r = \frac{n}{l}$). The probability of capturing t sensor nodes belonging to the same cluster is $P_t = \frac{l.C_t^r}{C_t^n}$. Figure 11 shows the probability of capturing t sensor nodes belonging to the same cluster for different threshold values. We can observe that this probability drops to zero very quickly after increasing the threshold value. Figure 12 shows this probability for a different number of clusters. Again, we observe that this probability drops to zero after increasing the number of clusters. Results show that our proposed scheme has probabilistic security and the probability of sensor capture attacks can be effectively reduced to be almost nonexistent by increasing the threshold $t(t \geq 3)$ or the number of clusters l (*i.e.*, $l \geq 4$).

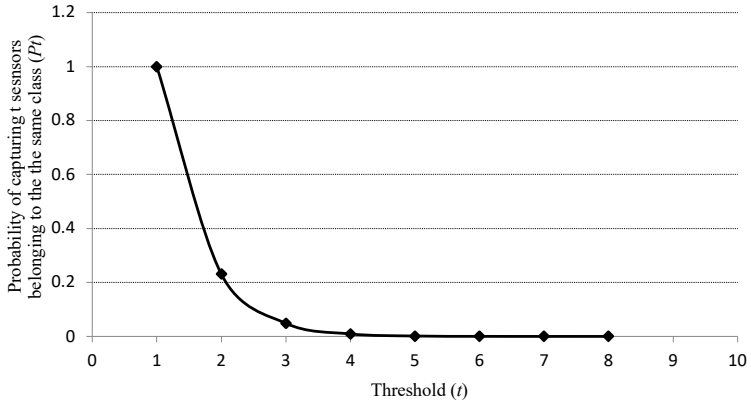

Figure 11: The probability of capturing t sensors belonging to the same cluster with various thresholds (for $l = 4, n = 40, r = 10$).

4.1.5.5 Comparison

Table 4 compares our proposed scheme with schemes in [46], [47], [50]. Our proposed scheme stores two polynomial shares in each sensor. The degrees of the polynomials are $h - 1$ and $t - 1$; thus each sensor stores $(t + h)$ coefficients. Compared to [46], each sensor node stores a master key and a univariate polynomial of degree R, which means that each sensor stores $R + 1$ coefficients. In [47], CHs generate shares to each sensor in its cluster, which are bivariate polynomials. In addition, each sensor is preloaded with a secret key. In [50], each sensor stores three keys: a network key that is used to secure communication between sensors; a cluster key that is used to secure communication between a sensor and its cluster head; and a share of a secret that is assumed to be utilized for group communication. Each of the keys and the share is of size p,

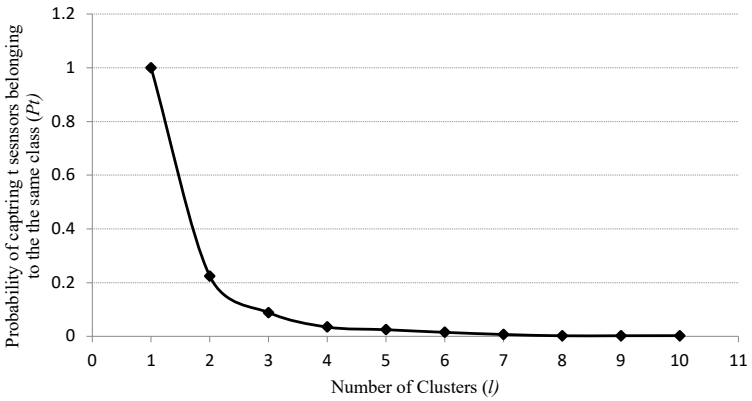

Figure 12: The probability of capturing t sensors belonging to the same cluster with various numbers of clusters (for $n = 30, t = 3$).

which is the modulus size in the scheme, requiring a storage space of three p-bit keys. For computation overhead, our proposed scheme only needs to do a polynomial evolution, whereas in [46], [47], each sensor requires decrypting the message, computing a hash, and doing a polynomial evaluation. In [50], sensors are preloaded with the required keys and they do not need to do any computation to establish the keys. Our proposed scheme has low communication overhead since each sensor is preloaded with the shares and there is no need to transmit any data other than the sensor's ID, which does not produce an overhead. On the other hand, the key establishment schemes in [46], [47] require many messages to be exchanged to authenticate sensor nodes and distribute polynomials used for generating the shared key, which shows a high communication overhead. In [50], the scheme shows high communication overhead due to the dynamic change of thresholds by the BS changes, which leads to reconstructing new shares for sensors nodes.

The problem with all deterministic key establishment schemes, including [46], [47], [50], is that their security is deterministic, so an attacker can successfully reconstruct the polynomial used to generate tokens after capturing t sensors and that compromises the security of the whole network. However, our proposed scheme is the first to provide probabilistic security for deterministic polynomial-based key establishment schemes, in which capturing more than t sensor nodes belonging to the same cluster has a very low probability, as explained in the security analysis section. In [50], if an attacker compromises a sensor, the attacker can obtain not only the network key that leads to compromising all data transmitted between sensors but also the cluster key that allows for the capture of all data transmitted between the compromised sensor and its cluster head. Thus, capturing one sensor leads to compromising the security of the whole WSN.

Table 4: Comparing the proposed scheme with other schemes.

Comparison Criteria	Proposed Scheme	[46]	[47]	[50]
Storage overhead	Two univariate polynomials $(t+h)$	A univariate polynomial $(R+1)$	Bivariate polynomial Secret Key	3 p-bit keys
Computation overhead	Polynomial evaluation	Polynomial evaluation Encryption/Decryption Hash function	Polynomial evaluation Hash function	-
Communication overhead	Low	High	High	High
Security against sensor-captured attack	Probabilistic	Deterministic	Deterministic	Deterministic

4.2 part 2: Non-Interactive Group Key Pre-Distribution Scheme (GKPS) for End-to-End Routing in WSNs

Most existing key distribution schemes in WSNs enable two sensors to establish a pairwise shared key which is generated and pre-loaded to the sensors by a key generation center (KGC) before deploying them into an area. The pairwise key is utilized to encrypt and authenticate data transmitted between two sensors. In a communication path, which involves multiple links, the key establishment is executed repeatedly in every link to route encrypted data successfully. Recently, a novel design of a secure end-to-end routing protocol [37] has been proposed based on a group key pre-distribution scheme (GKPS). The group key, also called a *path key*, is used to protect data transmitted in the entire path. Thus, instead of using multiple pairwise shared keys in a link-to-link secure communication, it uses an end-to-end path key, protecting data over the entire path. It can be concluded that implementing the end-to- end protocol is far more efficient and secure than the link- to-link protocol [37].

In this work, we propose a novel design of GKPS, which is based on a multivariate polynomial, but the security of our scheme is *probabilistic k-secure*. It is probabilistic to compromise the security of our proposed GKPS after capturing $k + 1$ or more sensors. We show that the probability of sensor capture attacks can be significantly reduced. Furthermore, our GKPS is very flexible in establishing path keys in WSNs. We need to point out that if the token of each sensor is stored without any tamper-resistant technology, it is quite easy for the attacker to recover the token of the sensor. In other words, it is quite impossible to prevent the attacker from recovering the token of that captured sensor

without employing any tamper-resistant technology. In this work, our proposed scheme does not prevent such an attack since we do not employ any tamper-resistant hardware. On the other hand, since our scheme is probabilistic k-secure, then after capturing $k+1$ or more than $k+1$ sensors, the attacker has an extremely low probability of obtaining the secret polynomial used to generate tokens of all sensors. Our scheme can prevent the attacker from obtaining all tokens of sensors after capturing $k+1$ or more than $k+1$ sensors. Thus, our proposed scheme enhances the system's security. Our contributions are as follows [85]:

- We propose two group key pre-distribution schemes: a deterministic scheme and a probabilistic scheme.
- Both schemes are based on a multivariate polynomial but with limited storage requirements.
- The security of the second GKPS is probabilistic.

4.2.1 Model

4.2.1.1 Description of Proposed GKPS

In our proposed GKPS, sensors are divided into multiple classes. Each sensor has a unique token initially generated and pre-loaded by key generation center (KGC). The storage space of each sensor is linearly proportional to the number of classes and is independent of the number of sensors. In addition, this scheme allows multiple sensors to establish a group key (i.e., also called **"path key"** in [37]) non-interactively. Figure 13 shows different paths protected by different group keys to securely routing the data

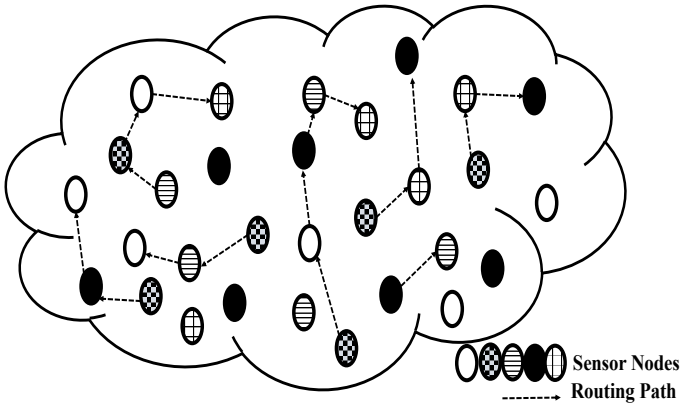

Figure 13: Group keys are utilized to securely routing data between sensor nodes.

through the entire path from source to destination sensor nodes. By changing the number of sensors in a WSN, the probabilities of establishing path keys with different lengths change as well. Similarly, changing the number of classes in a WSN can also change the probability of connectivity. One unique feature of our proposed scheme is the fact that it is the first polynomial-based GKPS with probabilistic security.

4.2.1.2 Performance

The following definitions will be used to evaluate the performance of the proposed GKPS.

Definition 1 *(Probability of capturing t sensors belonging to the same class P_t): After capturing t sensors, this is the probability that all captured sensors belong to the same class.*

In Theorem 3, we show that this parameter can be used to determine the security

strength of our GKPS.

Definition 2 *(Deterministic k-secure GKPS): A GKPS is said to be k-secure if GKPS can resist attacks that capture up to k sensors.*

After deploying sensors in a WSN, attackers may try to capture sensors and recover secret tokens. The parameter k is used to evaluate the security strength of a GKPS and its ability to resist such attacks. In most polynomial-based key distribution schemes, adjusting the degree of the polynomial is the only way to defend against a sensor capture attack.

Definition 3 *(Probabilistic k-secure GKPS): A GKPS is said to be probabilistic k-secure if GKPS can resist an attack by capturing up to k sensors. Furthermore, after capturing k+1 sensors or more, it is probable that the adversary can successfully compromise the security.*

Most existing polynomial-based key distribution schemes are deterministic k-secure schemes. To elaborate, capturing $k + 1$ or more than $k + 1$ sensors enables the adversary to successfully compromise the security of the network. Regardless, our GKPS is a probabilistic k-secure scheme. It is probabilistic that the adversary can successfully compromise the security. One of our design goals is to lower this probability in order to strengthen the security of the scheme. Our GKPS is very flexible since changing parameters of the GKPS can effectively lower this probability.

Definition 4 *(j-length GKPS): A GKPS is said to be j-length if GKPS can establish a group key among j+1 sensors.*

Most key establishment schemes in WSNs can only establish a pairwise secret key

between two sensors. One unique feature of our proposed GKPS is that it can establish a group key among multiple sensors, so data can be protected by a path key [37]. A path key with length j involves $j + 1$ sensors. So, the collected data in a WSN can be routed and protected by a path key. The parameter j is determined by many factors, such as the geographic size of the WSN, the total number of sensors, and the transmission distance of each sensor. In our proposed GKPS, we can adjust this parameter j, in order to facilitate an end-to-end secure communication.

Definition 5 *(Connectivity): Sensors are said to be connected to each other if any two sensors share a common secret key.*

Connectivity is a property of a WSN that determines whether information can be securely transmitted within a WSN. A deterministic key establishment scheme guarantees a shared pairwise key between any two arbitrary sensors. Thus, a deterministic key establishment ensures a connected network. On the other hand, probabilistic key establishment schemes, such as the random key scheme [29], does not guarantee a pairwise key between each pair of sensors within the network. As a result, such probabilistic key establishment schemes do not necessarily guarantee connectivity. When evaluating the schemes, the probability of connectivity is a parameter used to evaluate the performance of a probabilistic GKPS. In our proposed GKPS, we can increase the probability of connectivity by adjusting the parameters of the scheme.

Definition 6 *(Probability of connectivity with path length j P_j): the probability that any j+1 sensors can establish a group key (path length is j).*

In most key establishment schemes, pairwise keys are used to protect transmitted

data. The path length of these schemes is always restricted to equal one . However, our proposed GKPS can establish group keys with different path lengths. The parameter, P_j, is the probability of successfully establishing a path key involving $j+1$ sensors. In the performance section, we will discuss how to adjust this parameter.

Definition 7 *(Probability of connectivity P_c): the probability that any two sensors can establish a shared key.*

This parameter is the probability that data can be protected and transmitted securely in a WSN.

4.2.2 Proposed Schemes

In this section, we propose different schemes: the basic scheme, a modified scheme, and the proposed GKPS in detail.

4.2.2.1 Basic Scheme

We assume that there are l sensors, $S_i, i = 1, 2, \ldots, l$.

(a) Token Generation

The key generation center (KGC) needs to select l different polynomials, $f_i(x_i), i = 1, 2, \ldots, l$,and use them to generate tokens for sensors. Each polynomial is a univariate polynomial having $t-1$ degree. The token for each sensor, S_i, is $T_i = f_i(ID_i) \prod_{j=1, j\neq i}^{l} f_j(x_j) \bmod N$, where ID_i is the public information of a sensor, S_i, and N is the RSA modulus [86] which is the product of two large primes, p and q.

(b) Group Key Establishment

The group key, $K = \prod_{j=1}^{l} f_j(ID_j) \bmod N$, shared among l sensors, $S_i, i =$

$1, 2, \ldots, l$, can be computed by each sensor, S_i, using its secret token, T_i, and other sensors' IDs by computing $K = f_i(ID_i) \prod_{j=1, j \neq i}^{l} f_j(ID_j) \bmod N$.

Remark 1. The basic scheme can not only establish the group key for l sensors, $S_i, i = 1, 2, \ldots, l$, but can also establish group keys for any k (i.e., $2 \leq k \leq l$) sensors. For example, the group key among sensors, $S_i, i = 1, 2, \ldots, k$, is $K = \prod_{j=1}^{k} f_j(ID_j) \prod_{j=k+1}^{l} f_j(0) \bmod N$.

(c) Example 1

Assume that there are 3 sensors, S_1, S_2, and S_3. The KGC will select 3 polynomials, $f_1(x_1), f_2(x_2), f_3(x_3)$,and generate the tokens, $f_1(ID_1)f_2(x_2)f_3(x_3)$ for S_1, $f_1(x_1)$ $f_2(ID_2)f_3(x_3)$ for S_2, and $f_1(x_1)f_2(x_2)f_3(ID_3)$ for S_3, where ID_i is the public information of S_i. Note that each polynomial evaluation is computed using a RSA modulus N. As a result, a group key, $f_1(ID_1)f_2(ID_2)f_3(ID_3)$, can be shared among the 3 sensors, S_1, S_2, S_3, and a pairwise key can also be shared between any two sensors. For example, the key,$f_1(ID_1)f_2(ID_2)$

$f_3(0)$, can be shared between S_1, S_2.

(d) Security

We need to point out here that after capturing one sensor by the attacker, it is quite easy to recover the secret token of the sensor if the token is stored without using any tamper-resistant technology. In our proposed scheme, since we do not employ any tamper-resistant hardware, our scheme cannot prevent such an attack. On the other hand, our scheme is *probabilistic k-secure*, after capturing $k + 1$ or more than $k + 1$ sensors. In the following theorem, we demonstrate how attackers have an extremely low probability

in obtaining the secret polynomial used to generate tokens of all sensors.

Theorem 1. *The adversaries cannot obtain any information of secret polynomials selected by KGC.*

Proof. In the analysis of the sensor capture attack, we classify the attacks into two types.

1. *Capturing one sensor*- It is obvious that by capturing any single sensor S_i , and obtaining the token $T_i = f_i(ID_i) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod N$, the adversary cannot recover information of any individual polynomial, $f_i(x_i), i = 12, \ldots, l$, nor the product of all individual polynomials, $\prod_{j=1}^{l} f_j(x_j) \bmod N$.

2. *Capturing all sensors*- Assume that l sensors, $S_j \in c_j, j = 1, 2, \ldots, l$ (all sensors belong to different classes) with their public IDs, $ID_j \in, j = 1, 2, \ldots, l$, respectively, have been captured by an adversary. Then, multiplying their tokens $T_j = f_j(ID_j) \prod_{i=1,i\neq j}^{l} f_i(x_i) \bmod N, j = 1, 2, \ldots, l$, the adversary can obtain the product $\prod_{i=1}^{l} f_i(ID_i)(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$. Consequently, the adversary can remove $\prod_{i=1}^{l} f_i(ID_i)$ from the product and obtain $(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$. Next, the adversary may try to substitute $x_i = ID_i', i = 1, 2, \ldots, l$, where ID_i's are identities, into the polynomial $(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$, and gets $(\prod_{i=1}^{l} f_i(ID_i)')^{l-1} \bmod N = K^{l-1} \bmod N$. Based on the RSA assumption in [86], it is computationally infeasible to solve K. On the other hand, it is computationally impossible to solve the $(l-1)$-th root of $(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$ to obtain the secret product of polynomials,$(\prod_{i=1}^{l} f_i(x_i)) \bmod N$.

In the basic scheme, each sensor, S_i, needs to store a token, $T_i = f_i(ID_i) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod N$, which is a product polynomial of $l-1$ univariate polynomials where each

individual polynomial has degree $t-1$. Each sensor stores t^{l-1} coefficients of the product polynomial in Z_N. The storage space is exponentially proportional to the number of sensors. The following modified scheme can be used to reduce storage from exponential complexity to linear complexity.

4.2.2.2 Modified Scheme

This section explains a modified version of the basic scheme aimed at reducing the storage requirements from exponential complexity to linear complexity.

(a) Token Generation

The KGC follows the same procedure to generate tokens for all sensors as described in the basic scheme. The token for each sensor S_i , is $T_i = f_i(ID_i) \prod_{j=1,j\neq i}^{l} f_j(x_j) \mod N$. In addition, for each token, the KGC will randomly selects $l-1$ secret integers $a_j \in Z_N, j = 1, 2, \ldots, l, j \neq i$, such that $f_i(ID_i) = a_1 a_2 \ldots a_{i-1} a_{i+1} \ldots a_l \mod N$, and uses them to divide the token, $T_i = f_i(ID_i) \prod_{j=1,j\neq i}^{l} f_j(x_j) \mod N$, into $l-1$ sub-tokens, $s_{i,j} = a_j f_j(x_j), j = 1, 2, \ldots, l, j \neq i$. Note that the multiplication of all sub-tokens $\prod_{j=1,j\neq i}^{l} s_{i,j} = \prod_{j=1,j\neq i}^{l} a_j f_j(x_j)$, can recover the original token, $T_i = f_i(ID_i) \prod_{j=1,j\neq i}^{l} f_j(x_j) \mod N$. Each sensor is pre-loaded with sub-tokens, $s_{i,j} = a_j f_j(x_j)$, $j = 1, 2, \ldots, l, j \neq i$.

Since each sub-token is a univariate polynomial, storage of each sensor is the coefficients of $l-1$ univariate polynomials. In other words, the total storage of this modified scheme is $t(l-1)$; which results in a linear complexity.

Theorem 2. *The security of the modified scheme is the same as the basic scheme.*

Proof. For each sensor S_i, it stores $l-1$ sub-tokens, $s_{i,j} = a_j f_j(x_j), j = 1, 2, \ldots, l, j \neq i$. Since $l-1$ integers, $a_j \in Z_N, j = 1, 2, \ldots, l, j \neq i$, are randomly selected by the KGC for every sensor, it is computationally impossible to recover any individual polynomial, $s_{i,j} = a_j f_j(x_j), j = 1, 2, \ldots, l, j \neq i$, from its sub-tokens. The only information available when capturing any sensor is obtaining the token that provides the same knowledge obtained when capturing a sensor in the basic scheme.

4.2.2.3 Proposed GKPS

In most sensor network applications, a large number of sensors has to be deployed in order to cover a wide geographical area. If the number of sensors, n, are too large, it is impractical to implement the above modified scheme since it requires a large storage space of each sensor (i.e., the storage is $t(n-1)$).

(a) Token Generation

The KGC evenly divides n sensors into l classes $c_i, i = 1, 2, \ldots, l$, and each class, c_i, is associated with a distinct polynomial, $f_i(x_i)$, with degree $t-1$ each. Tokens of sensors in the same class are generated by the KGC using the same formula but with different IDs. For example, for two sensors, $S_1$and $S_2 \in c_i$, with $ID_{i,1}$ and $ID_{i,2}$, respectively, the tokens are $f_i(ID_{i,1}) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod N$, and $f_i(ID_{i,2}) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod modN$, respectively. For token, $T_{i,1} = f_i(ID_{i,1}) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod N$, the KGC will randomly select $l-1$ integers, $a_j \in Z_N, j = 1, 2, \ldots, l, j \neq i$, such that $f_i(ID_{i,1}) = a_1 a_2 \ldots a_{i-1} a_{i+1} \ldots a_l \bmod N$, and use them to divide the token, $T_{i,1} = f_i(ID_{i,1}) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod N$, into $l-1$ sub-tokens, $s_{i,j} = a_j f_j(x_j), j = 1, 2, \ldots, l, j \neq i$. Note

that the multiplication of all sub-tokens, $\prod_{j=1,j\neq i}^{l} s_{i,j} = \prod_{j=1,j\neq i}^{l} a_j f_j(x_j)$, enables the recovery of the original token, $T_{i,1} = f_i(ID_{i,1}) \prod_{j=1,j\neq i}^{l} f_j(x_j) \bmod N$. Sub-tokens, $s_{i,j}, j = 1, 2, \ldots, l, j \neq i$, are stored in sensor S_1.

(b) Group Key Establishment

Multiple sensors, which belonging to different classes can establish a group key as described in the basic scheme. However, sensors belonging to the same class cannot establish a group key. For example, we consider l sensors with their public IDs, $ID_j, j = 1, 2, \ldots, l$,respectively. If $S_j \in c_j, j = 1, 2, \ldots, l$, then the shared group key is $\prod_{i=1}^{l} f_i(ID_i)$, which can be computed by all sensors in the group. On the other hand, if there are only two sensors, S_{l-1} and S_l, in the subset of sensors, $\{S_j, j = 1, 2, \ldots, l\}$, belonging to the same class (i.e., $S_j \in c_j, j = 1, 2, \ldots, l-2$, and $S_{l-1}, S_l \in c_{l-1}$), then the shared group key among $l-2$ sensors belonging to different classes, $S_j \in c_j, j = j = 1, 2, \ldots, l-2$, is $f_{l-1}(0) f_l(0) \prod_{i=1}^{l-2} f_i(ID_i)$.

(c) Security Analysis

This section demonstrates the security analysis of the proposed GKPS against sensor capture attacks.

Theorem 3. *The proposed GKPS can resist attacks in capturing up to $t-1$ sensors in which all captured sensors should belong to the same class.*

Proof. In the analysis of the sensor capture attacks, we classify the attacks into two types:

1. *All captured sensors belong to the same class*- Assume that t sensors, $S_j \in c_1, j = 1, 2, \ldots, t$, (all sensors belong to the same class c_1) with their public IDs, $ID_j, j =$

$1, 2, \ldots, t$, respectively, have been captured by an adversary. Then, following Lagrange interpolation on these tokens, $T_j = f_1(ID_j) \prod_{i=2}^{l} f_i(x_i) \bmod N, j = 1, 2, \ldots, t$, the adversary can obtain the product of secret polynomials, $(\sum_{j=1}^{t} f_1(ID_j) \prod_{i=1, i\neq j}^{t} \frac{x_1 - ID_j}{ID_i - ID_j}) \prod_{i=2}^{l} f_i(x_i) \bmod N = \prod_{i=1}^{l} f_i(x_i)$, necessary to break the security of our proposed GKPS. Note that the adversary can only obtain the product of all existing polynomials but cannot obtain nor produce the individual polynomials. However, if the number of captured sensors is equal to or fewer than $t - 1$, the adversary cannot obtain the product of all individual polynomials, $\prod_{i=1}^{l} f_i(x_i)$, since the degree of each individual polynomial is $t - 1$. Furthermore, all captured sensors need to be in the same class in order for the Lagrange interpolation to work properly.

2. *All captured sensors belong to different classes*-Assume that l sensors, $S_j \in c_j, j = 1, 2, \ldots, l$ (all sensors belong to different classes) with their public IDs, $ID_j, j = 1, 2, \ldots, l$, respectively, have been captured by an adversary. Then, from Theorem 1, by multiplying their tokens, $T_j = f_j(ID_j) \prod_{i=1, i\neq j}^{l} f_i(x_i) \bmod N, j = 1, 2, \ldots, t$, the adversary can obtain the product, $\prod_{i=1}^{l} f_i(ID_i)(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$. By removing $\prod_{i=1}^{l} f_i(ID_i)$ from the product, the adversary can get $(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$. Regardless, it is computationally impossible to solve for the $(l-1)$-th root of $(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$ and obtain the secret product of the polynomials, $(\prod_{i=1}^{l} f_i(x_i)) \bmod N$. On the other hand, the adversary may try to substitute $ID'_i, i = 1, 2, \ldots, l$, where $\forall ID'_i \notin \{ID_i, i = 1, 2, \ldots, l\}$, into the polynomial, $(\prod_{i=1}^{l} f_i(x_i))^{l-1} \bmod N$, to get $(\prod_{i=1}^{l} f_i(ID'_i))^{l-1} \bmod N = K^{l-1} \bmod N$, where

ID_i' are identities of the group key, K. Nonetheless, based on the RSA assumption [86], it is computational infeasible to get the key, K.

Remark 2. Note that the sensor capture attack as described in the aforementioned theorem can only be applied if all captured sensors are in the same class. This condition increases the difficulty of sensor capture attack since captured sensors are randomly distributed in WSNs. In summary, our proposed GKPS effectively reduces the risk of a sensor capture attack since this attack only works if the following two conditions are satisfied simultaneously: (a) having captured t or more than t sensors; and (b) among all captured sensors, there must exist at least t sensors belonging to the same class. In the performance analysis section, we prove that being able to capture sensors from the same class has a very low probability.

Remark 3. If we limit the number of sensors in each class to be less than or equal to the degree of each individual polynomial (i.e., $\lfloor \frac{n}{l} \rfloor \leq t-1$), then the sensor capture attack described in Theorem 3 can never endanger the security of our proposed GKPS.

Remark 4. The degree of each individual polynomial determines the competence of the GKPS in resisting the sensor capture attack. If the degree of each polynomial is $t-1$, then the WSN can resist attacks capturing up to $t-1$ sensors in which all captured sensors belong to the same class. Increasing the degree of the polynomials can strengthen the security; but that increases the storage and computational requirements of the sensors (will discuss this in the next section.)

(d) Properties of group keys

1. *Non-interactive key establishment*- Sensors within a group can establish the group

key using its secret token and all other sensors' public identities. After forming the group, there is no need to exchange any information among sensors.

2. *Secrecy of group keys*- In our proposed GKPS, sensors in different classes can establish a group key. The group key is a function of the individual polynomials associated with classes and sensors' identities. Any sensor not belonging to the group cannot obtain this group key. For example, we consider l sensors, with their public IDs, $ID_j, j = 1, 2, \ldots, l$, respectively. If $S_j \in c_j, j = 1, 2, \ldots, l$, then the shared group key is $\prod_{i=1}^{l} f_i(ID_i)$, which can be computed by all the sensors in the group. On the other hand, for any other sensor, $S_1' \in c_1$, with ID_1', computing the group key from its token, $f_1(ID_1') \prod_{i=2}^{l} f_i(x_i)$ is not feasible.

3. *Key independence*- Each group key is a function of individual polynomials associated with classes and sensors' identities. Thus, each group key is independent of other group keys. However, if an attacker compromises t or more than t group keys belonging to a special subset, the attacker can recover all other group keys. The following theorem describes this type of known group key attack.

Theorem 4. *Known group key attack- If t or more than t group keys in a special subset are compromised by the adversary, then adversary can use the compromised group keys to recover other group keys.*

Proof. We use the following example to describe this special subset of compromised group keys. We assume that the attacker has compromised t group keys, $K_j = f_1(ID_{1,j})$ $f_2(ID_2) \ldots f_l(ID_l), j = 1, 2, \ldots, t$. Note that in this special subset of group keys, there

is only one identity, $ID_{1,j}, j = 1, 2, \ldots, t$, which is a variable that represents t sensors belonging to the same class, c_1, whereas the rest of the identities are fixed values. Using Lagrange interpolating formula, the attacker can obtain $\left\{\sum_{j=1}^{t} f_1(ID_{1,j}) \prod_{i=1, i\neq j}^{t} \frac{x_1 - ID_{1,j}}{ID_{1,i} - ID_{1,j}})\right\}$ $\left\{\prod_{i=2}^{l} f_i(ID_i)\right\} \bmod N = f_1(x_1) \prod_{i=2}^{l} f_i(ID_i)$. The attacker then can use this result to compute other group keys, $K_i = f_1(ID_{1,j}) f_2(ID_2) \ldots f_l(ID_l), \forall j, j \neq 1, 2, \ldots, t$.

Remark 5. The probability of this type of known group key attack is extremely low since it requires all captured group keys to belong to a special subset of group keys. Furthermore, the usefulness of this attack is very limited since the recovered keys must belong to a special subset of group keys as well.

4.2.3 Performance

In this section, we demonstrate the security analysis and performance analysis in terms of storage, computation, and connectivity of our proposed scheme. First, let us define the notations used in the section:

n: number of sensors in WSN

l: number of classes of sensors

$t - 1$: degree of each individual polynomial

m: number of sensors in each class (i.e., $m = \lfloor \frac{n}{l} \rfloor$)

4.2.3.1 Security

From Theorem 3, our proposed GKPS is a probabilistic $(t - 1)$-secure GKPS. In other words, our scheme can resist attacks of capturing up to $t-1$ sensors. After t or more than t sensors are captured, if and only if all captured sensors belong to the same class can

the adversary successfully compromise the GKPS. Concluding, the ability of an adversary to compromise the security of our scheme is proven probabilistic upon the preceding assertions. If t sensors within a network are captured, the probability that all captured sensors belong to the same class (P_t) is $P_t = \frac{C_i^m.l}{C_i^n}$. Figure 14 exhibits the probabilities of P_t for varying numbers of sensors. In this analysis, it is proven that as network size increases, the probability of capturing t sensors belonging to the same class increases. However, the increases in probability are quite small (i.e., $P_t = 0.00006367$ for $n = 120$) and can almost be disregarded. Thus, a sensor capture attack will not affect security if the network size is increased. Figure 15 shows the probabilities of P_t for a different number of classes within a network. The figure exhibits that increasing the number of classes can significantly decrease the probability of capturing t sensors belonging to the same class. In Figure 16, the probability P_t is sharply decreased with large thresholds (i.e., $t \geq 4$). From these results, it is proven that increasing either the number of classes, l, or the threshold value, t, can effectively lower the probability P_t. This result demonstrates that our GKPS is very flexible to enhance the security of the polynomial-based key distribution scheme. The design objective is to lower this probability P_t as much as we can.

4.2.3.2 Storage requirements

In the proposed GKPS, each sensor needs to store $l-1$ sub-tokens, $s_{i,j} = a_j f_j(x_j)$, $j = 1, 2, \ldots, l, j \neq i$, where each $f_j(x_j)$ is a univariate polynomial having degree $t - 1$ with coefficients in Z_N. In other words, the storage requirement is $(l - 1)t$ coefficients in Z_N, which is a linear complexity.

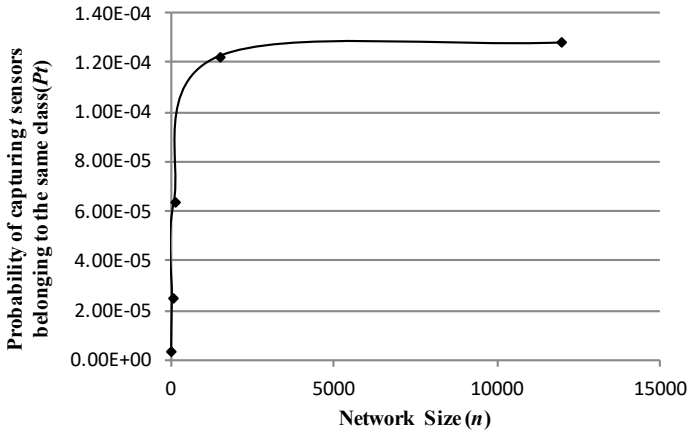

Figure 14: The probability of capturing t sensors that belong to the same class (P_t) with various number of sensors (for $t = 6$ and $l = 6$).

4.2.3.3 Computational requirements

We evaluate the computational effort to establish a path key with full length $l - 1$. Each sensor, S_i, must use its sub-tokens to compute $\prod_{j=1,j\neq i}^{l} a_j f_j(ID_j)$, where $ID_j, j = 1, 2, \ldots, l, j \neq i$. In other words, each sensors needs to evaluate $l - 1$ univariate polynomials with the degree $t - 1$. Each polynomial evaluation can follow Horner's rule [84] which requires $t-1$ multiplications and t additions. In total, each sensor needs to compute $(l - 1)(t - 1)$ multiplications in Z_N, which is a linear complexity.

4.2.3.4 Connectivity evaluation

We have proposed a probabilistic $(t - 1)$-secure $(l - 1)$-length GKPS. In general, parameters in our proposed GKPS are determined in the following manner. From

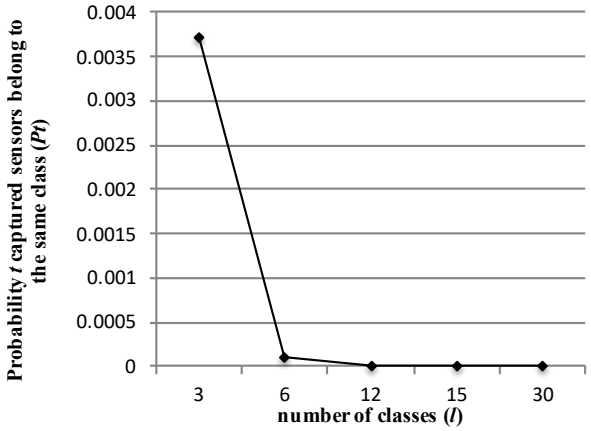

Figure 15: The probability of capturing t sensors belonging to the same class (P_t) while varying the number of classes (for $n = 300$ and $t = 6$).

the geographic size of a WSN and the communication distance of each sensor, the maximal length,$l - 1$, of a communication path in the WSN is determined first. Then, from the security requirement, the degree of each individual polynomial, $t - 1$, is determined. According to storage requirements of each sensor, we need to select sensors capable of storing at least $(l - 1)t$ integers in Z_N. Finally, the number of sensors, n, can be properly determined in order to provide adequate connectivity and satisfactory probability for establishing a path with a certain length.

1. *Probability of connectivity with path length j-* A path with length j must involve $j + 1$ sensors. In our proposed GKPS, these $j + 1$ sensors must all belong to different classes so that a group key can be established. The property P_j can be computed using the following formula, $P_j = \frac{C^l_{j+1} . m^{j+1}}{C^n_{j+1}}$.

   Figure 17 shows the probabilities of $P_j, j = 1, 2, 3, 4, 5$, for the total number of

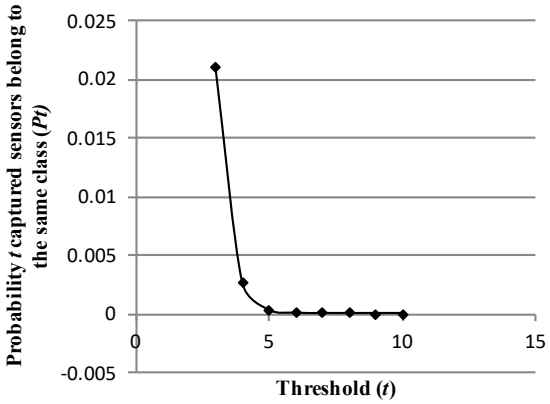

Figure 16: The probability distribution of all sensors belonging to the same class (P_t) when t sensors are captured; with various threshold values (for $n = 60$ and $l = 6$).

classes, $l = 6$. The ability of our GKPS to establish varying-length path keys is a unique feature in our GKPS that distinguishes it from most pairwise key establishment schemes, in which paths are bounded by the length 1. Fig. 17 shows the probabilities P_j for different numbers of n. When the length of the path increases, the probability is gradually decreases. Moreover, the probability of connectivity between any two sensors is very high (i.e. 83%) and gradually decreases as the path length is increases. In addition, increasing the number of sensors in the network can only slightly affect the probability of connectivity. Therefore, we are able to increase the size of the network covering the entire geographical area and almost get the same connectivity as that of a smaller network. Note that increasing n will not affect the storage requirements of sensors.

2. *Probability of connectivity*- In our proposed GKPS having l classes, if two sensors

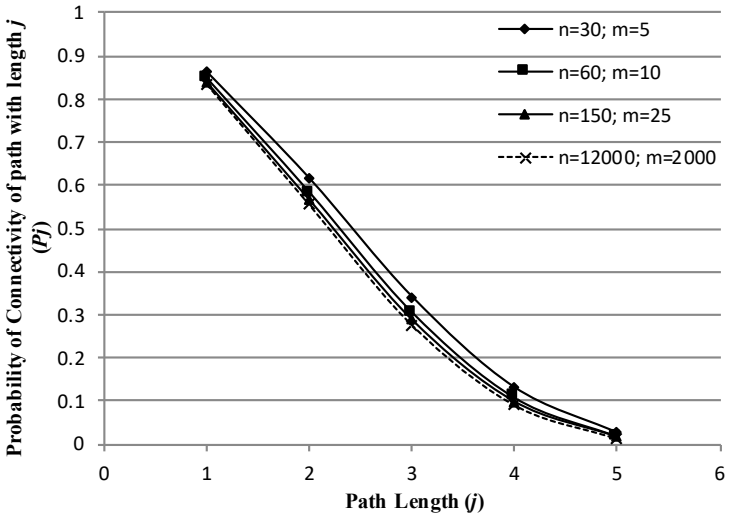

Figure 17: Probability distribution of connectivity with path length j.

belong to different classes, these two sensors are connected; otherwise, they are disconnected. The probability of dis-connectivity (P'_c) is: $P'_c = \frac{l.C_2^m}{C_2^n}$; and the probability of connectivity (P_c) is: $P_c = \frac{m^2.C_2^l}{C_2^n} = 1 - P'_c$.

One possible way to increase the probability, P_c, is to increase the number of classes in the WSNs. If the number of sensors, n, is fixed, increasing the number of classes, l, causes the number of sensors in each class to decrease. As a result, the probability that two sensors belong to the same class decreases, which in turn increases the probability of connectivity. Figure 18 shows this probability P_c for different numbers of classes, l.

Note that if $l = n$, then each sensor belongs to a unique class. This situation is

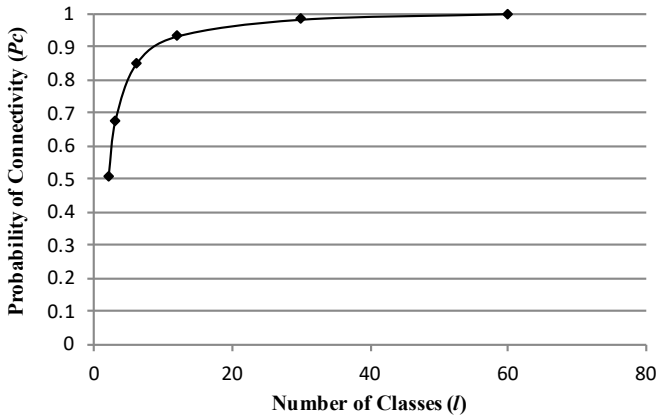

Figure 18: The probability distribution of network connectivity with different number of classes (for $n = 60$).

identical to the basic scheme, which is a deterministic key establishment scheme with $P_c = 1$. Notably, increasing the number of classes can increase both storage and computational costs of each sensor. This observation explains the motivation of our proposed GKPS, which is a flexible scheme since it can adjust parameters properly to balance the needs of high connectivity, strong security, and proper storage requirement for each sensor.

In case we need to deploy a large number of sensors to cover a large WSN, Figure 19 shows this probability P_c for different numbers of sensors. Although this result shows that increasing the size of a network can slightly decrease the probability of connectivity, the probability, P_c, remains very high for large number of sensors (e.g. $P_c = 0.83, n = 1200$). This result demonstrates the merits of our proposed GKPS. It can provide a high probability of connectivity for a wide range of sensors.

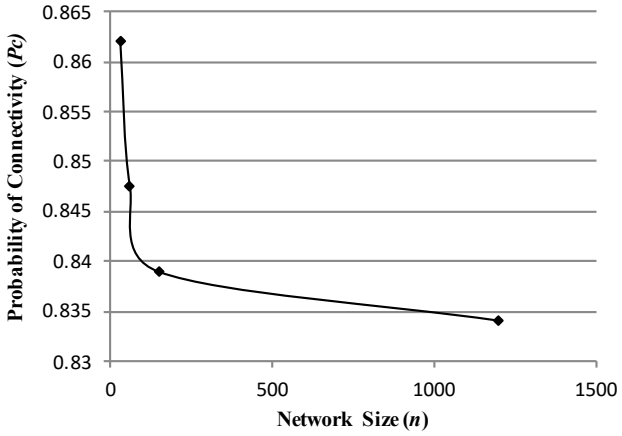

Figure 19: Probability distribution of network connectivity with different number of sensors.

4.2.4 Comparison

In this paper, we proposed two GKPSs to establish "path keys" among sensors in order to facilitate secure communications within sensor networks. Data transmitted over a communication path do not need to be protected using multiple pairwise shared keys in a link-to-link transmission. Instead, data can be protected by a single path key using an end-to-end transmission. Our proposed GKPS is a probabilistic $(t-1)$-secure$(l-1)$-length GKPS. Thus, our scheme's security is effectively strengthened compared to the deterministic schemes proposed in [54] [55] . We summarize the comparison with other key establishment schemes in Table 5.

Table 5: Comparison among Different Key Establishment Schemes

Schemes	Security assumption	Type of keys	Type of key establishment	Sensor capture attack
Random Key [29]	No assumption	Pairwise shared key	Probabilistic	Probabilistic secure
Group Key [54, 55]	RSA	Group keys	Deterministic	Deterministic k-secure
Basic scheme	RSA	Group keys	Deterministic	Deterministic k-secure
Proposed GKPS	RSA	Group keys	Probabilistic	Probabilistic k-secure

4.3 Conclusion

In this chapter, we proposed two novel polynomial-based key distribution schemes with probabilistic security in WSNs: the hierarchical key management scheme and the group key pre-distribution scheme (GKPS). The proposed schemes are *deterministic* key distribution schemes which guarantee that a pairwise key is shared between any two arbitrary sensor nodes in the WSNs, thus achieving high connectivity network. The *probabilistic* security feature of the proposed schemes has a huge impact in strengthen the security of the WSNs against sensor-captured attacks. In addition, both proposed schemes require sensor nodes of minimal memory, communication, and computational overhead.

CHAPTER 5

FOG COMPUTING

Low-cost microprocessors, sensors and wireless technologies lead to emerge of various physical devices designed to serve specific applications such as smart watch, smart lightbulb, smart thermostat, etc. These devices communicate with other devices as well as connect to the cloud in which user can control the devices remotely; these devices are called Internet Of Things (IoT). According to Cisco [9], IoT devices are going to increase rapidly and it is anticipated that 50 billion IoT devices will be connected to the Internet by 2020. The accretion of IoT devices connected to the cloud has created several network issues such as high latency, high bandwidth and network congestion due to massive amount of data sent to the cloud [87].

To overcome the aforementioned issues in the cloud, Cisco delivered the vision of fog computing [88], also called *Edge Computing*. Fog computing is an extension of the cloud that is deployed at the edge of network near users. The IoT devices will be connected to the fog instead of the cloud, and the IoT devices will not be aware of this transition. With Fog computing, IoT will receive better service in terms of low latency, and high response rate which is critical for time-sensitive applications [89].

Security services such as data confidentiality and data authentication are needed in any network communication such as fog computing to protect the transmitted data from various kind of attacks. To provide security services in fog, users and fog nodes (*i.e.*, base

station, router, server, etc.) must share a key which is used to protect users' data, this is where Key *Establishment/Distribution* comes into light.

IoT devices have limited capabilities in terms of memory storage, computational power and battery life [90]. Thus, designing a security protocol in such resource-constrained devices should consider these limitations. The existing key distribution schemes in computer networks are not applicable in IoT due to inherit characteristics of IoT devices such as limited memory, computation and battery power. Public-Key schemes are impractical in IoT because they require large key size and high computational power [90]. Symmetric cryptographic scheme is the suitable choice for IoT because it has small key size (*i.e.*, 128-bit for AES compared to 1024-bit for RSA) and less computational power compared to public-key schemes. There are several symmetric key establishment schemes used for resource-limited devices as explained in [91]. One of the symmetric schemes is a polynomial-based scheme in which devices are preloaded with polynomial coefficients and use them to create pairwise keys that are used to secure the data transmitted through the public channel, *the Internet* [37].

In this chapter, we propose a lightweight polynomial-based key management scheme in Fog Computing. Our scheme has a *hierarchical* key management structure, in which fog nodes are deployed at the end of the network near user devices; and keys are generated based on the hierarchical structure. In summary, our scheme has the following features:

- It guarantees a shared key between a fog node and every End-User (EU) nodes.
- It provides various keys to support secure fog-to-cloud, fog-to-fog and fog-to-user communication.

- It's a scalable, lightweight and hierarchical key management structure that can accommodate a large number of EU nodes without increasing the storage requirements of EU nodes.

5.1 System Model

Fog computing acts as an intermediate layer between the cloud and the user devices. This layer shapes the cloud structure to appear as a hierarchy with the cloud in the top of the hierarchy and the user nodes in the bottom of the hierarchy. The fog structure is depicted in Figure 20. The bottom of the hierarchy encompasses various End-User (EU) devices that range from infrastructure like home and buildings to heterogeneous IoT devices such as cameras and vehicles, in which those devices are equipped with sensing and communication capabilities to sense environmental phenomenon and send such data through communication channel to fog node to do further processing and analysis on the collected data. The system entities are explained below:

- *Key Distribution Center (KDC)*: it is responsible for registering each device that joins the fog network and then generating tokens for all nodes in the system (*i.e.*, fog nodes and user nodes). It is located in the cloud data center; and once it generates all tokens, it goes offline.

- *Fog node (FN)*: it is located in the fog layer between the cloud and the user node layer. It is the critical component that acts as a relay between the cloud and the user nodes. Each fog node serves EU nodes within its communication range (*i.e.*, fog cluster). Fog nodes can be a base station, server or router.

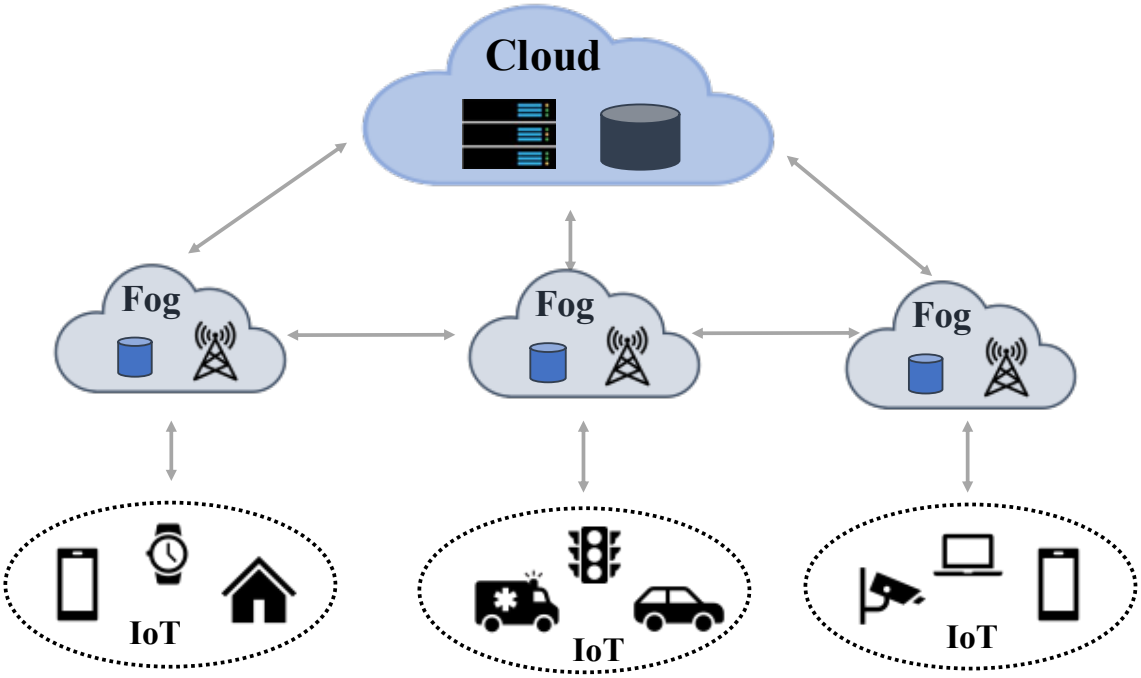

Figure 20: Fog structure.

- *The cloud*: contains data centers, KDC and the control center that receives all data from End-User nodes and process and analyze them according to some application requirements.

- *End-User (EU) nodes*: there are various types of EU devices present in a fog cluster that is served by a fog node. Fog layer has many fog clusters that are deployed in a geographical area in which fog nodes in a specific fog cluster serve all EU nodes in its cluster. Due to heterogeneous user devices, we divide user nodes based on their types to smart phones, vehicles, homes, etc.

5.2 Proposed Scheme

In our proposed scheme, key generation is done in two phases: *registration* and *key establishment*.

5.2.1 Registration

Each EU node must be registered with the KDC, which is responsible to assign a unique ID and tokens to each node (*i.e.*, FNs and EUs) in the network. Tokens in our proposed hierarchical key management scheme are generated in top- down approach. The KDC selects a trivariate polynomial,$F(x, y, z) = a_{0,0,0} + \cdots + a_{0,1,1}yz + a_{1,0,0}x + a_{1,0,1}xz + a_{1,1,0}xy + a_{1,1,1}xyz + \cdots + a_{k-1,t-1,h-1}x^{k-1}y^{t-1}z^{h-1} \bmod p$, which is used to generate tokens for all nodes in the network. KDC uses its trivariate polynomial to issue tokens, which are bivariate polynomial,$f(ID, y, z)$, to all FNs. Thus, the token of FN with identity ID_i is $f(ID_i, y, z) = f_{ID_i}(y, z) \bmod p$. In the same way, KDC uses the bivariate polynomial,$f(ID, y, z)$, of each FN to generate tokens, which are integers, $f(ID, Type, id)$, to all EU nodes within its fog cluster. So, the token of the EU node with identity, id_j, that belongs to the fog cluster of the FN with identity, ID_i, is, $f(ID_i, Type, id_j)$, where *Type* represent the EU device type. EU nodes are classified based on their types to cell phones, vehicles, buildings, etc. Each upper level node in the hierarchical network is able to access tokens of its lower level nodes. For example, the fog node in the fog layer knows all tokens of its EU nodes in the bottom layer. We explain the token generation in detail in the following subsection.

5.2.2 Token generation

A Key Distribution Center (KDC) initially selects a prime modulus, p, and a trivariate polynomial, $F(x, y, z)$, where p satisfies the cryptographic key's size that is sufficient to secure network communication; the degree of x is $k-1$, the degree of y is $t-1$, and the degree of z is $h-1$; all polynomial's coefficients are in $GF(p)$.

Token of each fog node with identity, ID_i, is a bivariate polynomial,$F(ID_i, y, z) \bmod p$. Each unique bivariate polynomial is kept by each fog node. Tokens of all End-User (EU) nodes in the same fog cluster are generated by the fog node (FN)'s bivariate polynomial, $F(ID_i, y, z) \bmod p$, where ,ID_i, is the FN's ID (we call it FN_i). The token of a EU node with identity, $id_j \in FN_i$, and type $Type$ (*e.g.*, cellphone, vehicle, building, etc.) is $F(ID_i, Type, id_j) \bmod p$, which is an integer that represent the pairwise shared key between EU node and its FN. The registration phase algorithm is explained below. Once all EU and FN nodes are registered and are loaded with its tokens by KDC, KDC goes offline.

Algorithm 1 Registration Phase

```
1:  procedure REGISTRATION(FN, EU)
2:      for each FN ∈ network do
3:          ID_i = FN.identity
4:          FN.token = f(ID_i, y, z) mod p
5:      for i ← 1 to n do            ▷ n is the number of registered FNs in the network
6:          for each EU ∈ FN_i do
7:              ID_i = FN.identity
8:              id_j = EU.identity
9:              Type = EU.device − type
10:             EU.token = f(ID_i, Type, id_j) mod p
11:     KDC goes offline
```

5.2.3 KeyEstablishment

Our scheme supports two types of communications: *unicast* communication, which is a one-to-one way of communication, and *broadcast* communication, which is a one-to-many way of communication. In the unicast communication, an EU node can send data to its fog node, or a fog node can communicate with other fog nodes in the fog layer. In the broadcast communication, the fog node can send data to all EU nodes in its fog cluster. The following subsections demonstrate the key establishment process in detail.

5.2.3.1 Unicast keys

- *EU-to- FN pairwise key*: Any FN can use its bivariate polynomial to generate a pairwise key and share it with EU nodes in its fog cluster. For example, the FN with identity, ID_i, and its bivariate polynomial, $F(ID_i, y, z) \bmod p$,can share the key,$K_{F,U} = F(ID_i, Type, id_j) \bmod p$, with the EU node with identity, id_j, and device type is $Type$ which can be any category of IoT device (*i.e.*, vehicle, smart phone, etc.).

- *FN-to- FN pairwise key:* If a fog node i wants to communicate with other fog node j, it needs to send a request to KDC. KDC will generate a key, $k_{i,j}$, and send it securely to the FNs with ID_i and ID_j. To elaborate, KDC will send two messages that include the key, $k_{i,j}$, encrypted with the pairwise key between KDC/Cloud and the FNs; the first message,$E_{K_{F_i,C}}(k_{i,j})$, is sent to FN with ID_i, where $K_{F_i,C}$ is the shared pairwise key between FN with ID_i and the KDC in the cloud. The second message,$E_{K_{F_j,C}}(k_{i,j})$, is sent to FN with ID_j, where $K_{F_j,C}$ is the shared pairwise

key between FN with ID_j and the KDC. Once FNs received the message, each FN uses its pairwise key to decrypt the message and retrieve the key, $k_{i,j}$, that will be used to secure the communication between the FNs with ID_j and ID_i.

- *FN-to- Cloud pairwise key:* The KDC in the cloud can use its trivariate polynomial, $F(x, y, z) \bmod p$, to share a pairwise key with any FN in fog network. For example, the KDC can share the key, $K_{F,C} = F(ID_i, ID_i, ID_i) \bmod p$, with the FN with identity, ID_i.

5.2.3.2 Broadcast key

A broadcast key is used by FN to send a secure broadest message to all EU nodes in its cluster. First, FN chooses a broadest key, K_B, and sent it to each EU nodes separately as, $E_{K_{F,U}}(K_B)$, in which the broadcast key, K_B, is encrypted under the shared pairwise key, $K_{F,U}$, with each EU node.

5.3 Security Analysis

In this section, we analyze the security of our scheme against possible attacks in fog environment.

5.3.1 Insider attack

Insider attack is an attack carried by an insider who has access to legitimate EU node and try to compromise the security of the local fog network by reconstructing the secret bivariate polynomial that is used to create all tokens in the fog cluster. This attack

works when several malicious insiders *collude* together and share their tokens to reconstruct the bivariate polynomial of the fog node.

To recover the token of the fog node, which is a bivariate polynomial, $F(ID_i, y, z) \mod p$, in which the degree of y is $t-1$ and the degree of z is $h-1$, there must be h or more than h colluded attackers to cooperate to release their shares and construct the bivariate polynomial. However, with less than h compromised EU nodes, attackers do not have any information about the bivariate polynomial and could not reconstruct the fog node's secret token. For example, let h malicious insiders share their tokens of their EU devices, which are all devices from the same type,$Type_j$, (*e.g.*, all are cell phones, or vehicles); then, they can reconstruct the fog node's bivariate polynomial using Lagrange Interpolation [65], $\sum_{j=1}^{h} F(ID_i, Type_j, id_j) \prod_{w=1, w\neq j}^{h-1} \frac{x-id_w}{id_j-id_w} \mod p = F(ID_i, y, z)$. However, with fewer than h tokens, attacker cannot recover the bivariate polynomial. By increasing the parameter h, the security of the system can be enhanced. The properties of the bivariate polynomial with different degrees can be found in [81].

5.3.2 Impersonation attack

In this type of attack, an attacker tries to pretend to be a legitimate user to access the network and collect the transmitted data. Since an attacker does not have a valid token generated from the KDC, the attackers attack attempt can be detected and excluded from the network activities. On the other hand, if attacker compromises one of the devices that is loaded with a valid token then attacker can get the shared pairwise key; however, this only compromises the link between the attacked device and the fog node but can not affect

other non-compromised links.

5.4 Performance Analysis

The performance of our proposed scheme is analyzed in terms of storage, communication and computation capabilities of EU nodes.

5.4.1 Storage complexity

In our proposed hierarchical scheme, the KDC stores the trivariate polynomial, $F(x, y, z)$, where the degree of x is $k-1$, y is $t-1$, and z is $h-1$. Thus, KDC needs to store kth coefficients in $GF(p)$. Each fog node with identity, ID_i, needs to store a bivariate polynomial, $F(ID_i, y, z) \bmod p$. The storage requirement of each fog node is th coefficients in $GF(p)$. Each EU node with identity id_j and $Type$, store an integer,$F(ID_i, Type, id_j) \bmod p$, which its size is equivalent to the modulus p. Consequently, our proposed hierarchical scheme minimized the storage in of EU nodes in the fog network; EU nodes at the bottom of the hierarchy store minimal coefficients compare to the KDC at the top of the hierarchy which stores the maximum number of coefficients. The storage of KDC and fog nodes are much higher than EU nodes, so the hierarchical key distribution scheme assigns keys in an efficient way.

Overall, our proposed scheme is scalable, more EU nodes can join the fog network without affecting the storage capacity of the existing EU nodes or the new joined nodes since in the EU node level, all devices need to store a key which is the size of the modulus *independent* of the network size. The fog node has an enough storage capacity to store the bivariate coefficients and to accommodate more IoT devices. Furthermore, KDC can

increase the degree of the trivariate polynomial to increase the security level and that can be tolerated at fog node layer without increasing the storage at EU nodes. Thus, the security can be enhanced without affecting the EU node's storage.

5.4.2 Computational overhead

In the following discussion, we evaluate computational requirements of our three types of communication keys. Horner's rule [84] can be used to reduce the computational cost in the polynomial evaluation. According to Horner's rule, evaluating a univariate polynomial of degree $h-1$ needs $h-1$ multiplications and h additions.

- *FN-to-EU Communication Key:* FN with identity, ID_i, uses its bivariate polynomial, $F(ID_i, y, z) \bmod p$, to establish a pairwise key, $K_{F,U} = F(ID_i, Type_j, id_j) \bmod p$, with EU node with id_j and $Type$. The bivariate polynomial has $t-1$ degree in y and $h-1$ degree in z. thus, FN needs th multiplications and $th+t-1$ additions. On the other hand, EU nodes are only loaded with a single integer that represent the shared key, $K_{F,U}$

- *FN-to-FN and FN-to-Cloud Communication Keys:* When FN wants to communicate with other FN, it needs to send such request to the KDC in the cloud, so it needs to establish the shared key between FN with, ID_i, and the KDC. To evaluate the FN's bivariate polynomial, $F(ID_i, y, z) \bmod p$, to establish the shared key, $K_{F,C} = F(ID_i, ID_i, ID_i) \bmod p$, FN needs th multiplications and $th+t-1$ additions. However, KDC uses its trivariate polynomial, $F(x, y, z)$, to establish the shared pairwise key, $K_{F,C}$. The KDC needs $k(th+1)$ multiplications and

$kth + kt - 1$ additions.

5.4.3 Communication overhead

The proposed scheme has *low* communicational overhead. After deployment, no information needs to be transmitted to establish the shared pairwise keys between fog node and EU nodes except the ID of EU nodes which is attached with each data sent from the EU nodes.

5.4.4 Comparison

We compare our proposed scheme with the proxy-based polynomial scheme proposed by Porambage *et al.* in [63]. Table 6 shows the performance analysis comparison. In terms of storage, our proposed scheme requires only the pairwise key to be stored in EU node; however, in [63], a sensor node stores its private key, n pre-installed shared keys with its neighboring proxies and k out of n DH-key shares required to construct the DH key. In terms of communication, our proposed scheme does not need any messages to be sent from EU node to establish the pairwise key; on the other hand, the protocol in [63] exhibit high communication overhead in sensor nodes since each sensor node needs to send encrypted messages containing its shares to its n neighboring proxies, and receives at least k encrypted messages that include the DH- key shares and lastly send a MAC message to confirm to the host device the successful derivation of the DH key. In terms of computation, our proposed scheme does not require EU node to do any processing to establish the pairwise key since it is pre-installed. However, in [63], there are expensive computations required to establish the DH key which include polynomial computations

(*i.e.*, addition and multiplication) and exponentiation computation. Sensor nodes in [63] needs to encrypt n messages that include its shares and send them to its neighboring proxies and decrypt at least k out of n received encrypted messages that include the DH key shares; in addition, it needs to do k multiplications of the received DH-shares to construct the DH key and finally compute a MAC based on the DH key.

Table 6: Performance Analysis Comparison

Comparison Criteria	[63]	the Proposed Scheme
Storage overhead	$n + k + 2$ keys	1 key
Computation overhead	$n + k + 1$ encryption/decryption k-multiplications	0
Communication overhead	n+k+1 encrypted messages	0 messages

5.5 Conclusion

We proposed a polynomial-based hierarchical key management scheme in fog computing with deterministic key establishment that guarantees a shared key between a fog node and all user nodes in its cluster. Our scheme can resist up to h colluded attacks. With fewer than h insider attackers, information about the secret polynomial of degree $h - 1$ stored by fog node is not revealed. Furthermore, it is a scalable scheme in which more nodes can be added into the fog network without affecting the security of the network. In addition, the security level can be increased without increasing the storage requirements of the user nodes.

CHAPTER 6

BLOCKCHAIN NETWORKS

Since the emergence of Bitcoin, *the cryptocurrency*, blockchain technology has received substantial attention from academic, businesses, and governments. It has been applied in various environments such as smart cities [92], smart homes [16], and healthcare [14] to provide security services such as data integrity, confidentiality and system availability. Blockchain is a distributed ledger in a peer-to-peer network in which all transactions are stored in a chain of blocks. It uses cryptographic mechanisms to chain the blocks in a way that makes it difficult to manipulate the data stored in the ledger without being detected. Comparing with other traditional distributed systems, blockchain stands out for its three properties: trustless, permissionless and censorship resistance [93]. Blockchain's design ensures that no one entity can control data stored in the ledger; this provides the immutability of data and ensures all entities having almost the same copy of the ledger [94]. Furthermore, miners are entities who is responsible for validating transactions, reaching a consensus, creating blocks and updating the ledger. Validating a transaction triggers the execution of a smart contract (i.e., chaincode) which explains the rules to govern transactions [94].

Blockchain technologies are classified into two main categories: public (i.e., permissionless) and consortium (i.e., permissioned) blockchain [93]. Public blockchain allows any entity to join the network, read and write into the ledger. In addition, any entity

can participate as a miner, host the ledger, create a block and run the consensus processes. Due to its public nature, the mining process and the consensus algorithms are designed in a complicated way and requires an exhausted number of resources (e.g., processing power, computations, energy) to provide high immutability of distributed ledger and to increase the difficulty of attacks that try to manipulate the ledger. On the other hand, permissioned blockchain prefers identity over anonymity, which means it requires each entity to register prior joining the network. Thus, entities have some form of identifying each other but not necessarily trusting each other and making the collaboration in exchanging valuable assets and processing business possible without the need of a trusted third party. Furthermore, specific entities are assigned as miners which are the only entities who host the ledger, run consensus algorithms and are allowed to update the ledger. Comparing with the permissionless blockchain, the mining in permissioned blockchain is considered less expensive in terms of the consensus algorithms' complexity and number of resources (i.e., memory, processing and energy power) required to run mining processes. Although all blockchain ledgers offer the same security services overall, permissioned ledgers prohibit anonymous miners from validating transactions, and prevent potentially malicious sources from entering into the validation process. Therefore, only authorized miners may validate transactions within the system [95] [96], and that service maintains integrity throughout the validation process. Due to the aforementioned properties of permissioned blockchain, we consider the permissioned blockchain technology in our proposed scheme. Specifically, we adopt the hyperledger fabric architecture and communication flows [97] in designing our proposed scheme.

Generally speaking, to secure communication over public channel, such as the *Internet*, two main security services are required, user authentication and data confidentiality. To provide such services, each entity in the network needs to share a key with other entities before exchanging data over the network. This is called *key establishment*. In blockchain networks, we noticed that all existing key establishment schemes are based on the public-key infrastructure (PKI), which is an interactive key establishment that requires information to be exchanged and verified between entities in order to generate a key. This results in a long-time process of key establishment. Our proposed scheme is a pre-distribution scheme which means entities are preloaded with tokens that enable them to non-interactively share keys with other entities in the network. This results in much faster key establishment process comparing to PKI. In addition, public-key operations, such as the RSA modular exponentiation, have more modular multiplications than the polynomial computations.

In this Chapter, we propose a polynomial-based key management scheme in permissioned blockchain utilizing bivariate polynomials. We aim at reducing the amount of information that needs to be transmitted and stored in entities, while allowing each entity to be able to compute pairwise keys independently. Also, we explain the advantages of our proposed scheme as well as its limitations.

6.1 Background

6.1.1 Hyperledger Fabric Blockchain Architecture

The main components of the hyperledger blockchain network are peer nodes which can be classified into a validating peer and a non-validating peer. A validating peer is the node that hosts ledgers, runs smart contracts and endorses/validates transactions [98]. On the other hand, a non-validating peer is responsible to connect clients' applications to validating peers, but it does not host ledgers and can not execute transactions. The hyperledger fabric main entities are described below:

- Client's Application (A): It is the entity that request a transaction by sending a transaction proposal (TP) to endorsing peers, which are specified in the endorsing policy of the smart contract (SC). Later, client's application will receive endorsed transactions, called transaction proposal responses (TRs), from endorsing peers. Then, client's application checks that it receives sufficient number of TRs that need to be sent to an Orderer.

- Endorsing Peer (EP): It endorses transactions that comes from several clients' applications according to endorsement policy specified in smart contract of each transaction. In addition, EP must validate endorsed transactions that come from Orderer before applying them to the ledger.

- Orderer (O): Once Orderer receives TRs from clients' applications, it orders all

transactions, puts them in blocks and distributes blocks to all peers (EP and non-endorsing peers) in the blockchain network. Orderer is one of the important non-validating peer nodes that plays a significant role in hyperledger's finality because it prevents the "ledger fork" and that keeps ledger in consistent states among all peers [99].

6.1.1.1 General Communication Transaction Flow

From Figure 21 , the communication flow in hyperledger fabric blockchain follows five basic steps [99] [100]:

- Step 1: Client's application (A) sends a transaction proposal (TP) to a specific set of endorsing peers (EP) in the network (i.e. endorsing peer 1, endorsing peer 2 and endorsing peer N). Then, each EP executes a smart contract (i.e. chaincode) on the transaction proposal, endorses it and creates a transaction proposal response (TR).

- Step 2: Once endorsing peers create TRs, they send them back to the client's application. After receiving the required number (i.e., a threshold number) of endorsed TRs, client's application checks the results of the transaction proposal responses to ensure they have the same results.

- Step 3: Client's application sends all TRs to an Orderer. Assuming that there are multiple Orderers in the blockchain network, client's application specifies to which Orderers to submit TRs.

- Step 4: Orderer receives TRs from client's application, orders them according to

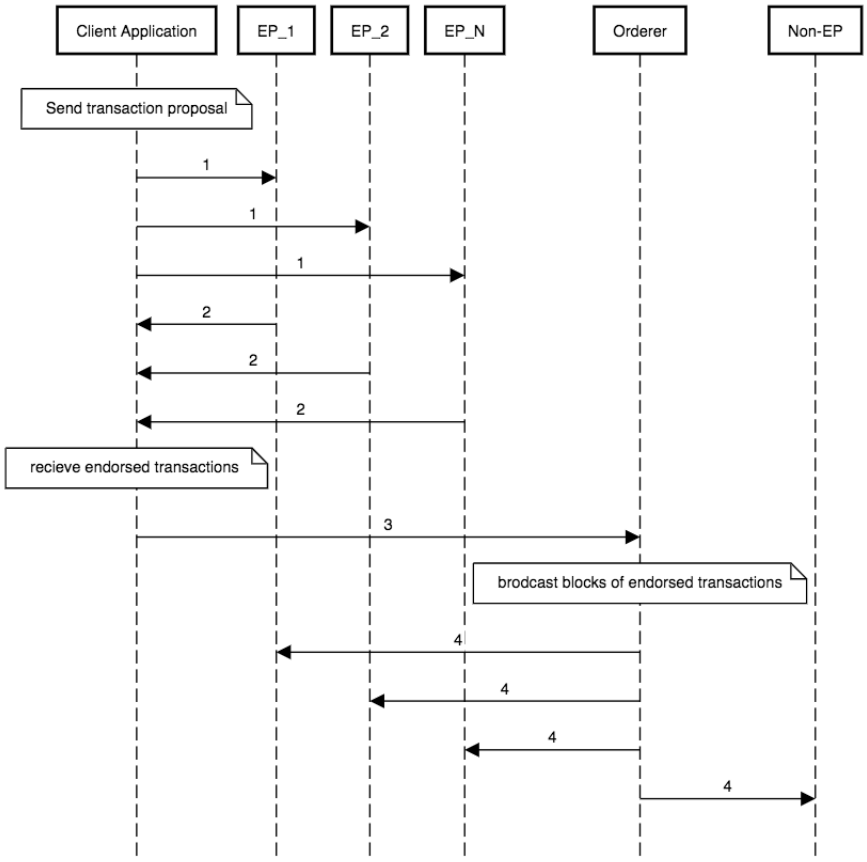

Figure 21: Transaction flow

other transactions that comes from other applications, puts them in blocks and distributes blocks to all peers in the blockchain network, including the endorsing peers and non-endorsing peers.

- Step 5: A committing peer must validate all transactions in the blocks to ensure that they have been consistently endorsed by all relevant endorsing peers according to the endorsement policy of the smart contract. Thus, only the validated transactions in the blocks will be applied to the ledger while the failed transactions will be

rejected.

6.1.1.2 Additional Communication Transaction Flow

In the blockchain network, it is common that some organizations want to privately share transactions with a subset of organizations in the network without exposing these transactions to the whole network. The hyperledger fabric satisfies such requests by providing two methods that ensure privacy of transactions between a subset of organizations in the blockchain network: the *channel* and the *private data collection* [101]. Channel is a private subnet of communication between two or more network members who establish the channel to conduct private and confidential transactions. Figure 22 shows blockchain with several channels. Only authorized peers are allowed to participate in channel's activities and share a private data.

However, creating multiple channels to protect data between any subset of network nodes (i.e., entities) leads to additional administrative overhead. With private data collections, entities can share transactions with all channels' members while keeping portion of transactions private. Consequently, private data collection allows a subset of network members on a channel to share private data without having to create a separate channel. There are two components of the collection- *the actual private data* and the *hash of the private date*. The actual private data that is stored in a private database on the peers (i.e., network member) and is shared between the authorized members in a channel through peer-to-peer protocol (i.e., gossip network). The hash of the private data which is endorsed, ordered and written to the ledger of every peer on the channel so that hash serves

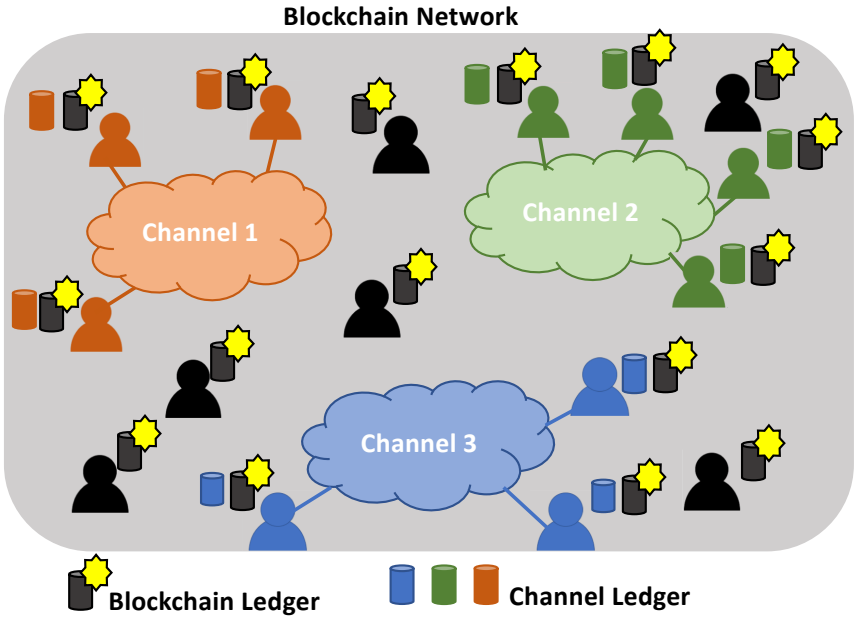

Figure 22: Blockchain network with several channels

as a proof of the transaction without revealing the private data in the channel [102].

All communication in the permissioned blockchain must be protected from outsiders and unauthorized entities. Thus, entity authentication and transaction confidentiality are required. In order to provide such security services, keys are needed to protect all transactions. As we mentioned earlier, most of the permissioned blockchain such as hyperledger fabric are based on Public-Key Infrastructure in which certificates generated from certificate authorities (CAs) are used to provide entity authentication and public/private key pair to encrypt/sign transactions between entities. All entities in the permissioned network need to authenticate other entities by checking their certificates

frequently, then get public keys to validate the endorsements (i.e., signatures) on the transaction according to endorsement policy coded in the smart contract (i.e., chaincode). Using public-key cryptosystem to authenticate entities and validate every transaction in the blockchain that encompasses large numbers of entities, require each entity to store other entities' certificates and do some computational-intensive operations to validate transactions. It needs a large storage to hold all certificates and more computational power to do all tasks in real time. In this Chapter, we propose a polynomial-based key management scheme for permissioned blockchains such as hyperledger fabric. Our scheme requires less storage and performs faster computations compare to PKI scheme.

6.1.2 Bivariate Polynomial Key Establishment Schemes

In [21], Blom proposed the first pairwise key establishment scheme based on a symmetric bivariate polynomial. In addition, in [7], Blundo *et al.* demonstrated the polynomial-based key establishment scheme in which a symmetric bivariate polynomial, $f(x, y)$, is used to generate the pairwise keys.

The Key Generation Center (KGC), which is responsible to generate keying materials for each entities in the network, selects a symmetric bivariate polynomial, $f(x, y) = a_{0,0} + a_{1,0}x + a_{0,1}y + a_{2,0}x^2 + a_{1,1}xy + a_{0,2}y^2 + \cdots + a_{t-1,0}x^{t-1} + a_{t-2,1}x^{t-2}y + \cdots + a_{t-1,t-1}x^{t-1}y^{t-1} \bmod p$, where the coefficients, $a_{i,j} \in GF(p)$, and $a_{i,j} = a_{j,i}, \forall i, j$. Then, KGC uses the bivariate polynomial, $f(x, y)$, to generate tokens, $f(ID_i, y), i = 1, 2, \ldots, n$, which are univariate polynomials and the ID_i is the public information of entity, $entity_i$. Since $f(x_i, x_j) = f(x_j, x_i), \forall i, j \in [0, t-1]$, the two entities with identities

ID_i and ID_j, can successfully create and share a pairwise key.

6.2 Model of the Polynomial-Based Key Management Scheme

In this section we introduce our scheme including the key establishment process and the attack model.

6.2.1 Key Establishment Process

Our proposed scheme consists of two phases: the token generation phase and the key establishment phase.

- Token Generation: The Key Generation Center (KGC) generates three bivariate polynomials, each aim at securing communication between two entities in the network. Then, KGC utilizes the bivariate polynomials to generate tokens, which are univariate polynomials, and preload tokens in entities.

- Key Establishment: after each entity are preloaded with tokens, they can use the tokens to create pairwise keys that is used to securely transmit data over the network.

6.2.2 Attack Model

There are two attack scenarios we consider in this paper: the insider attack and outsider attack. The *Inside attacker* is the one who possess a valid token and try to collude with other inside attackers to compromise the security of the network. On the other hand, *Outside Attacker* is an attacker who does not possess any valid token but tries to compromise an entity to either disrupt a service or obtain its valid tokens to compromise

network's traffic. Our proposed scheme can resist both attacks as we will show in the following sections.

6.3 Polynomial-Based Key Management Scheme

In this section, we propose a bivariate polynomial-based key management scheme in permissioned blockchain, more specifically, our solution is applied to the hyperledge fabric model. The scheme consists of three bivariate polynomials that support three types of communication. The first bivariate polynomial is created to support the two-way communication between client's application and endorsing peers. The second bivariate polynomial is created for the one-way communication between the client's application and an Orderer. The third bivariate polynomial is to support one-way communication between the Orderer and all peers in the blockchain network. Furthermore, our key management scheme can support both the general and additional communication flows, explained in previous section. The proposed scheme consists of two main phases, the token generation phase and the key establishment phase.

6.3.1 Token Generation Phase

Key Generation Center (KGC) creates three bivariate polynomials, $f_i(x, y) \bmod p, i = 1, 2, 3$, in which coefficients belong to $GF(p)$ where p is a prime modulus. The degree of x is $t - 1$ and y is $n - 1$. All entities in the blockchain network are assigned a unique identity number. Then, KGC uses each polynomial to generate tokens which are univariate polynomials for each entity as demonstrated below. The token generation phase is depicted in Figure 23 .

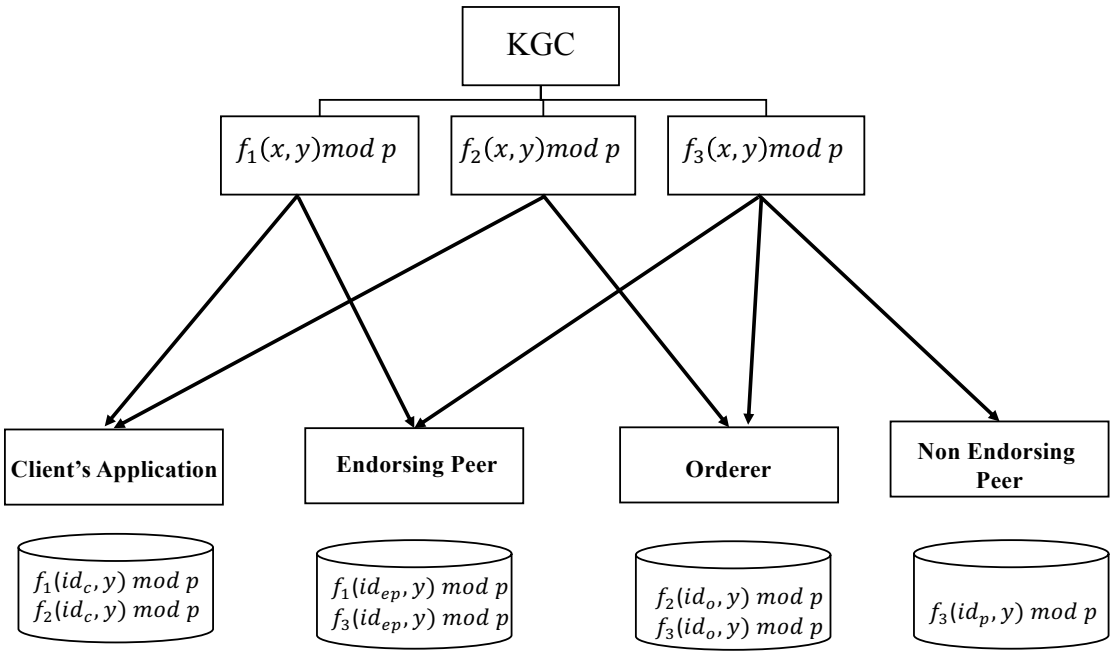

Figure 23: Token generation phase.

- For the two-way communication between a client's application and endorsing peers (EP), KGC generates tokens for client's application with identity, id_c, as $f_1(id_c, y) \bmod p$, and for EP with identity, id_{ep}, as $f_1(id_{ep}, y) \bmod p$.

- For the one-way communication between a client's application and Orderer, KGC creates tokens for client's application with identity, id_c, as $f_2(id_c, y) \bmod p$, and for Orderer with identity,id_o, as $f_2(id_o, y) \bmod p$.

- For the one-way communication between an Orderer and all peers (i.e., endorsing and non endorsing peers) in the network, KGC computes tokens for Orderer with identity, id_o, as $f_3(id_o, y) \bmod p$, and for a peer with identity,id_p, as $f_3(id_p, y) mod p$.

6.3.2 Key Establishment Phase

After KGC generates tokens for all entities in the blockchain network, each entity can use its tokens to compute the shared pairwise key with other communicating entities as the following:

- The shared pairwise key between a client's application with identity, id_c, and EP with identity,id_{ep}, is $f_1(id_c, id_{ep}) = f_1(id_{ep}, id_c) \bmod p = K_{c,ep}$, as seen in Figure 24 .

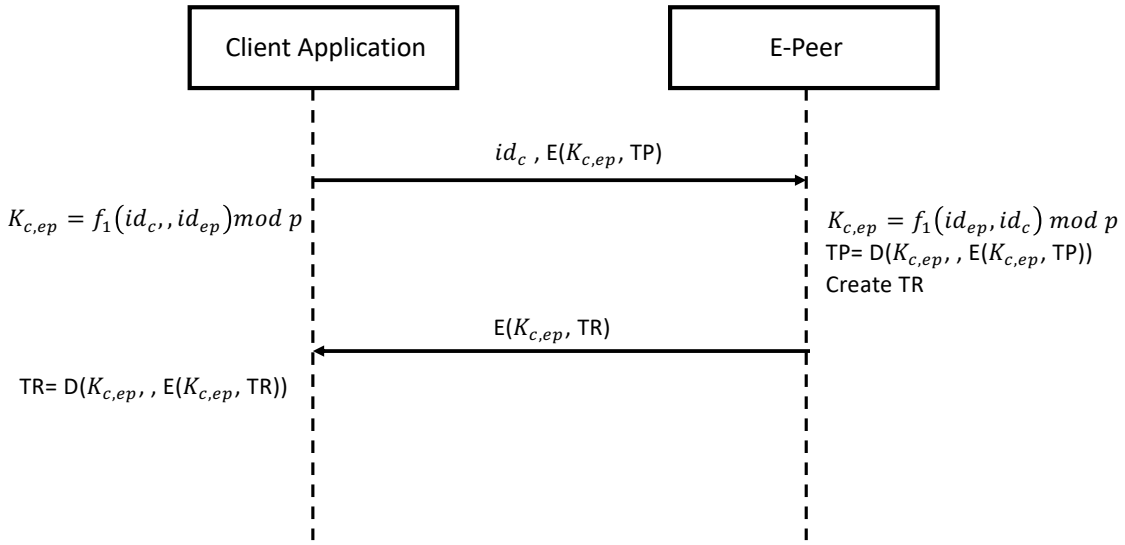

Figure 24: Key establishment between client's application and endorsing peer (EP).

- The shared pairwise key between a client's application with identity, id_c, and Orderer with identity,id_o, is $f_2(id_c, id_o) = f_2(id_o, id_c) \bmod p = K_{c,o}$, as shown in Figure 25 .

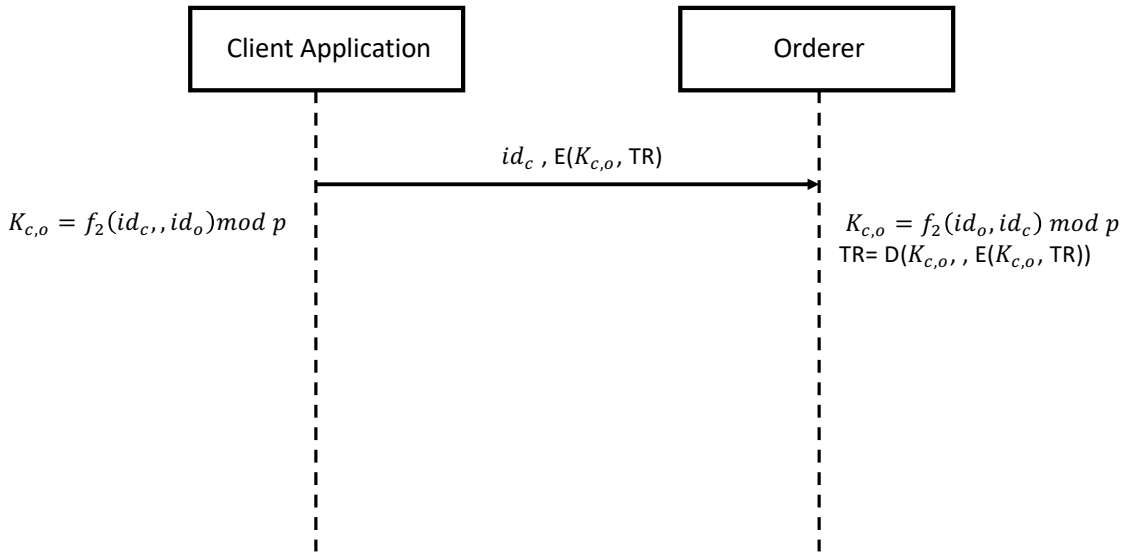

Figure 25: Key establishment between client's application and Orderer.

- Finally, The shared pairwise key between Orderer with identity, id_o, and a peers with identity, id_p, is $f_3(id_o, id_p) = f_3(id_p, id_o) \bmod p = K_{o,p}$, as depicted in Figure 26 .

Our proposed scheme also allows clients applications to communicate with each other using their tokens, $f_1(id_c, y)$ or $f_2(id_c, y)$. In addition, EPs can communicate with each other securely by generating a shared pairwise key from either $f_1(id_{ep}, y)$ or $f_3(id_{ep}, y)$. In the same way, Ordere can communicate with other Orderer using either $f_2(id_o, y)$ or $f_3(id_o, y)$ to generate a shared pairwise key.

6.4 Security Analysis

Having three independent bivariate polynomials to generate tokens for different type of communications, the security will be enhanced since compromising one bivariate

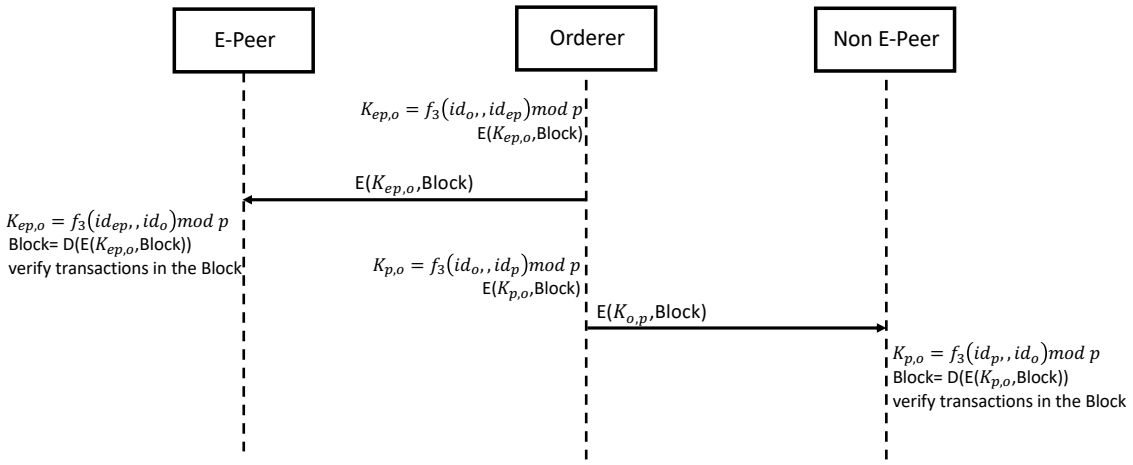

Figure 26: Key establishment between endorsing peer (EP), non endorsing peer and Orderer.

polynomial does not affect security of other bivariate polynomials corresponding to other types of communications.

6.4.1 Attack Model

We consider two types of attacks in our proposed scheme, the insider attack and the outsider attack.

6.4.1.1 Inside Attack

To recover one of the bivariate polynomials, $f_i(x, y) \bmod p, i = 1, 2$ or 3, used to generate tokens for entities, it takes at least t inside attackers to be colluded together to share their tokens and reconstruct the bivariate polynomial using Lagrange Interpolation as $(\sum_{i=1}^{t} f_k(ID_i, y) \prod_{j=1, j\neq i}^{t} \frac{x - ID_j}{ID_i - ID_j}) \bmod p = f_k(x, y)$,where $k = 1, 2,$ or 3. However, with fewer than t tokens, the bivariate polynomial cannot be obtained. For example,

when the inside attackers are clients' applications, then t clients' applications can collude together to recover the bivariate polynomial used to create the tokens between clients' applications and EPs. As a result, all secret tokens are recovered and can be used to sign and submit false transactions.

6.4.1.2 Outside Attack

If an attacker can compromise an entity and get hold of entity's tokens, the attacker can only affect the security associated with this compromised entity. However, if attacker has compromised t or more than t tokens of entities belonging to the same category (i.e., all entities belonging to either clients' applications, EPs or Orderers), attacker can reconstruct the bivariate polynomial used to create their tokens. This can lead to a serious security breach on the network security. However, it is very unlikely to compromise t entities especially when t is a large number. Intrusion detection and prevention system can be utilized to eliminate such attacks.

Note. *In the above two types of attacks, if we limit the number of entities whose tokens are generated from same bivariate polynomial of degree t to be fewer than t, then this attack can never be occurred.*

6.4.2 Security Enhancement Model

There are several security mechanisms that can be utilized to eliminate attacks in our proposed schemes. For example, tamper-proof mechanisms can be utilized to store the tokens securely in the entities. Thus, even though entities have been hacked, attacker will not be able to acquire the tokens. In other words, attacker will fail to reconstruct the

bivariate polynomial.

6.5 Performance Analysis

In this section, we discuss the performance of our proposed scheme in terms of storage, communication, and computational overhead.

6.5.1 Storage Requirement

Each entity needs to store tokens that are polynomial coefficients. In the case of a client's application, it stores two tokens, $f_1(id_c, y)$ and $f_2(id_c, y)$, corresponding to two univariate polynomials with degree $t - 1$ each. In other words, the client's application stores $2t$ coefficients. Also, EP stores two tokens, $f_1(id_{ep}, y)$ and $f_3(id_{ep}, y)$, so they are required to store $2t$ coefficients. In the same way, Orderer has two tokens, $f_2(id_o, y)$ or $f_3(id_o, y)$, and stores $2t$ coefficients in total. On the other hand, non-EP stores only one token, $f_3(id_p, y)$, and they only need to store t coefficients. In fact, number of coefficients stored in each entity is independent of network size. This property is attractive since it promotes scalability of the system as well as connectivity between entities.

6.5.2 Communication Overhead

Our proposed scheme does not require sensitive data (i.e., tokens) to be transmitted over the network. Only the identities of the communicating parties need to be exchanged to create the shared pairwise key. As a result, communication overhead is minimal in our proposed scheme.

6.5.3 Computational Overhead

The properties of the polynomial-based schemes make them better than the public-key schemes for the following reasons:

- The modulus size of the polynomial-based cryptographic scheme is much smaller than the modulus size required for the public-key cryptographic schemes. This is because the security of the polynomial-based cryptographic schemes is unconditionally secure, so the modulus size needs to be larger than the size of secret keys (i.e., 128-bit) of communication. On the other hand, the security of the public-key cryptographic schemes is based on some computational assumptions. For example, the security of the RSA scheme is based on the difficulty of factoring large composite integers (i.e., 2048-bit).

- The polynomial-based evaluation can be done following Horner's rule [84], which requires t modular multiplications, and t is the degree of the polynomial. In general, t is not a large integer (e. g., 10). However, the the computation of public-key cryptographic schemes such as the RSA scheme needs to evaluate modular exponentiations. Computing each modular exponentiation using *Squaring-and-Multiply* method [6] requires $1.5 \log_2 n$ modular multiplications. If n is 2048-bit, then it needs around 3000 modular multiplications.

- Thus, polynomial-based modular multiplication with a smaller modulus (i.g., 128-bit) is much faster than the public-key modular multiplication with a larger modulus (e.g., 2048-bit).

In the following paragraphs, we demonstrate the aforementioned properties by comparing our proposed scheme and PKI schemes, specifically RSA, both theoretically and experimentally.

6.5.3.1 Theoretical Analysis

We analyze the computational differences in terms of modular multiplications required for both the polynomial-based schemes and RSA cryptosystem. We utilized Horner's rule [84] for the polynomials' evaluation, and *Squaring-and-Multiply* algorithm [6] for RSA evaluation. In the polynomial-based scheme, a polynomial of degree t requires t number of multiplications [84]; in each multiplication, there is an x-bits number (i.e., x is the size of the modulus, which is the size of the secret key) multiplying by an x-bits number resulting in x^2 bit-level multiplications. So, the total number of bit-level multiplications required for the modular multiplications in the polynomial evaluations is $t \times x^2$. For instance, a polynomial of degree 10 with modulus size (x) 128-bit requires $163,840$ bit-level multiplications. On the other hand, the RSA modular exponentiation requires $1.5 \log_2 y$ modular multiplications. Each multiplication involves y^2 bit-level multiplications. This results in a total of $[y^2 \times (1.5 \log_2 y)]$ bit-level multiplications. For example, the total number of bit-level multiplications required for the modular exponentiation in the RSA with modulus size 1024-bit is 1.610613×10^9. We conclude that the RSA with modulus size 1024-bit is approximately 9830.4 times slower than polynomial-based scheme with degree 10 and modulus size 128-bit. Table 7 shows how much an RSA with different modulus sizes $(1024, 2048, 3072)$ is slower than the polynomial-based schemes

with various degree $(10, 50, 100)$ and modulus sizes $(128, 192, 256)$. [p]

We need to mention that a 2048-bit is the minimum key size for an RSA to be considered secure, since a 1024-bits RSA implementation was broken in 2010 [103]. On the other hand, the polynomial-based scheme is considered secure with smaller modulus sizes (i.e., 128-bit) and usually does not require large modulus sizes nor large degrees. Thus, due to larger modulus size requirements and more multiplication operations, RSA is considered far slower than the polynomial-based schemes to provide the same security services.

6.5.3.2 Experiments

We implemented both schemes, the polynomial-based scheme using Horner's rule [84], and the RSA modular exponentiation utilizing *Squaring-and-Multiply* algorithm [6]. We ran both algorithms in a 32-bit Ubuntu based Linux OS, on a VirtualBox installed in macOS Sierra, on a 64-bit laptop which has a 2-core Intel Core i7 $(2.9$ GHz), and 4 GB memory. We assigned one CPU and 1024-bits memory to the Ubuntu and we ran the algorithms 100 times. Figure 27 aims at showing the execution time in Nano seconds (*ns*) for polynomials with different degrees (e.g., $10, 50, 100$) and various modulus sizes (e.g., $128, 192, 256$ bits). With a polynomial of degree 10, increasing the modulus size resulted in a slight increase in time (i.e., ≈ 0.02 milliseconds (*ms*)) required to evaluate the polynomial. Increasing the degree of the polynomial (i.e., > 50) will gradually increase the time required to evaluate the polynomial when the modulus size is increased as well. Notably, the time it takes to evaluate a polynomial of degree 100 and a modulus size

Table 7: Theoretical Analysis shows how the RSA is slower than Polynomial-based Schemes

		Polynomial								
Modulus Size		128-bit			192-bit			256-bit		
Degree		10	50	100	10	50	100	10	50	100
RSA	1024	9830.4	1966.08	983.04	4369.066667	873.8133333	436.9066667	2457.6	491.52	245.76
	2048	78643.2	15728.64	7864.32	34952.53333	6990.506667	3495.253333	19660.8	3932.16	1966.08
	3072	265420.8	53084.16	26542.08	117964.8	23592.96	11796.48	66355.2	13271.04	6635.52

256-bit is approximately 2.9 *ms*, which is considered fast if we consider this a worst-case scenario. Figure 28 shows the execution time to compute the RSA modular exponentiation with various modulus sizes (e.g., $100, 200, 1024, 2048, 3072$ bits). As we mentioned earlier, for security purpose, RSA modulus sizes must be greater than 1024 bits. We have demonstrated several modulus sizes to show how that can affect the execution time of RSA modular exponentiation. Fig. 28 exhibits that increasing the modulus sizes results in a dramatic increase in the execution time of RSA modular exponentiation. For example, a 1024-bit modulus size takes 41.80 *ms*, 2048-bit takes 306.53 *ms* and 3072-bit takes 952.81 *ms*. In Figure 29, we aim at comparing the time it takes to evaluate a polynomial with different degrees and different modulus sizes with the RSA modular exponentiation with various modulus sizes. The graph shows how RSA takes long time to compute the modular exponentiations with larger modulus sizes. It shows that the RSA is much slower than the polynomial-based schemes. From Fig. 29, we see that the 2048-bit RSA takes 3.07×10^8 *ns* (i.e., ≈ 306.5390 *ms*) while polynomial with degree 100 and modulus size 128-bit takes around 883942 *ns* (i.e., ≈ 0.883942 *ms*). There is a huge difference in the execution time of both schemes, RSA takes around 305 *ms* more to compute the modular exponentiation.

6.6 Comparison and Limitation

Most of the blockchain implementations have adopted Public-Key Infrastructure (PKI) as a key management scheme. The main drawback of utilizing PKI is that it is an

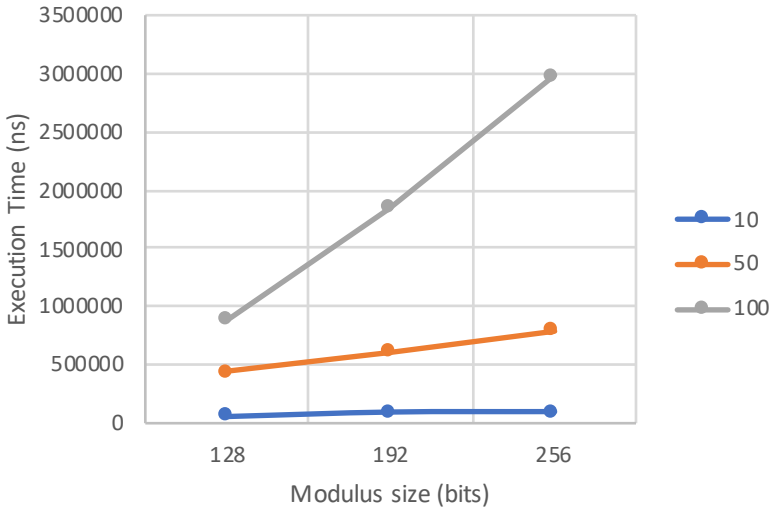

Figure 27: The execution time of evaluating polynomials with different modulus sizes (128, 192, 256 bits) and different degrees (10,50,100).

interactive scheme that requires validating information (i.e., certificates) exchanged between entities prior to establishing a shared key, decrypting a message, or verifying a signature. Thus, imposing every entity to check the validity of the certificate for every transaction creates an overhead and will cause a significant delay especially when the number of entities in the blockchain is very large. In addition, the operations involved in PKI such as validating certificates, encrypting/decrypting transactions, and signing/verifying the transactions are expensive in comparsion to the polynomial evaluation operations. In this paper, we propose a novel key management scheme in blockchain technology. The main advantage of our proposed scheme is that tokens are pre-loaded into each entity enabling them to generate shared keys. This can speed up the key establishment process significantly.

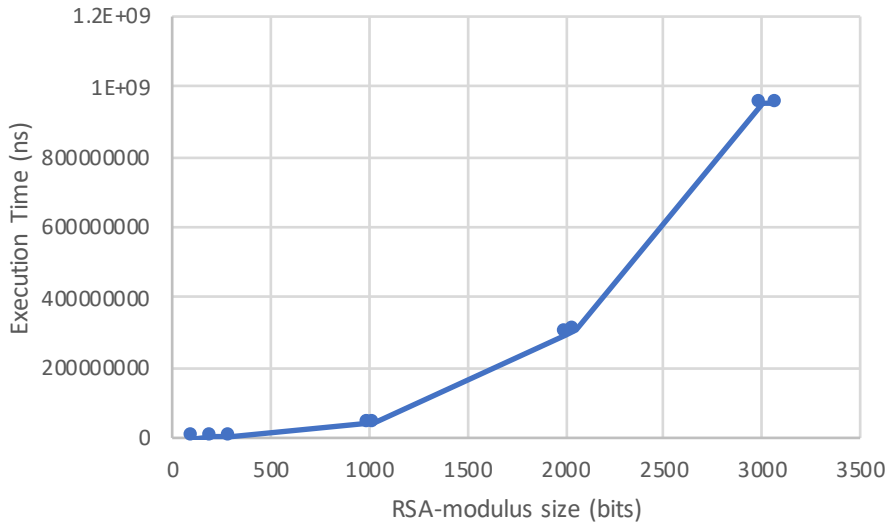

Figure 28: The execution time of computing modular exponentiation of RSA with different modulus sizes (in bits).

Table 8 provides a comparison between our proposed scheme and the PKI scheme in hyperledger fabric that is based on X.509 certificates and the Elliptic Curve Digital Signature Algorithm (ECDSA) [104]. In terms of storage requirements, our proposed scheme requires each entity to store two univariate polynomials' coefficients with a complexity of $O(t)$ where t is the degree of the polynomial. In PKI, each entity needs to store the valid certificates of other entities, $O(n)$ where n is the number of entities in the network. Therefore, the PKI requires more storage when more entities join the network (i.e., dynamic storage), in contrast to the polynomial-based scheme that stores polynomials' coefficients only and that does not affected if new entity join the network (i.e., static storage). Regarding the computations, our scheme requires polynomial evaluations that consist of

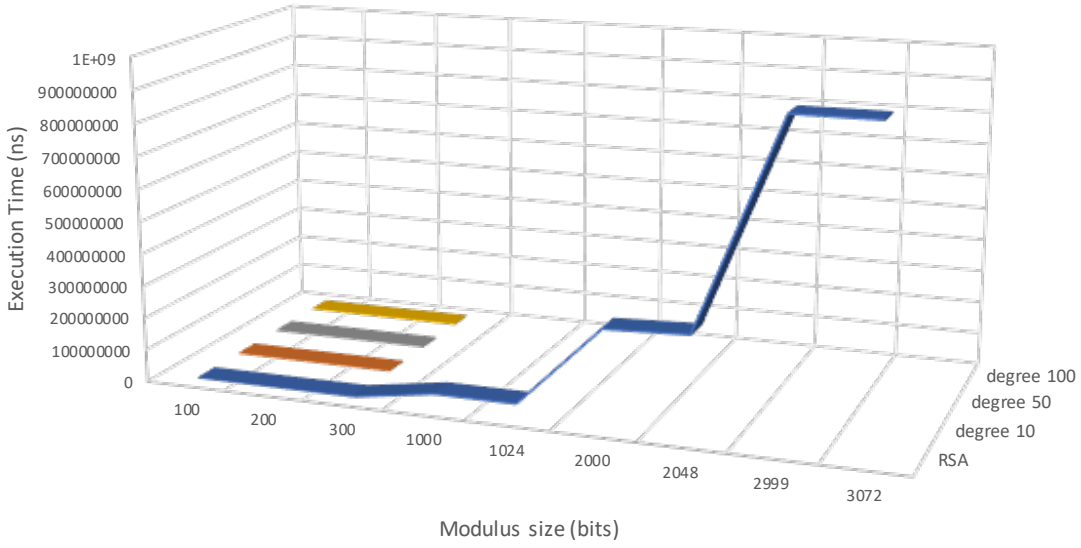

Figure 29: Comparison between the execution time of RSA modular exponentiation and polynomial-based evaluations with different modulus sizes (in bits).

primitive operations (i.e., multiplications and additions) while PKI requires expensive operations such as modular exponentiation that involves a large number of modular multiplications due to the large key size required for PKI security. For communication, entities in the proposed scheme exchange their IDs, which can be found in a public directory, to generate the shared keys where as in PKI, entities need to exchange/validate certificates before obtaining the public keys or calculating the shared keys. However, our proposed scheme lacks revocation mechanisms in case the shared keys are compromised by attackers. PKI, on the other hand, encompasses revocation mechanisms such as the Certificate Revocation List (CRL). This is considered a limitation in our proposed scheme and we are going to address it in our future work.

Table 8: Comparison between the Proposed Scheme and Public-Key Schemes

Entity's Characteristic	**Proposed Scheme**	**Public-Key Scheme** [104]
Storage	Polynomials' coefficients $(t \times x)$	Entities' certificates $O(ny)$
	Fixed storage	Dynamic storage
Independent from Network Size	Yes	No
Communication	Exchange IDs	Exchange Certificates
Computation	Polynomial Evaluation	Modular Multiplications
Revocation Mechanisms	No	Yes

6.7 Conclusion

In this paper, we propose a bivariate polynomial-based key management scheme in a permissioned blockchain. In our scheme, entities in the blockchain network are preloaded with tokens that enable them to share pairwise keys. Establishing pairwise keys involves less complex polynomial evaluation operations such as additions and multiplications, which result in a faster processing compared with the popular blockchain key management schemes such as public-key schemes. Moreover, our scheme is a lightweight key management scheme in which the entities would be required to store only tokens that are of linear complexity. In addition, the size of the stored tokens is independent of the network size and thus enhances the scalability of the network. The only limitation with our scheme is a lack of revocation mechanisms, and we plan to address this limitation in our future work.

CHAPTER 7

CONCLUSION AND FUTURE WORK

Resource-constrained devices such as IoTs and sensors become an integral part of most networked systems. These devices help users to access their data anywhere anytime. For example, a user can use their mobile phone to access their bank accounts and medical records, or to control their home appliances remotely. The data transferred from these devices contains sensitive information that must be protected before being transferred over networks; otherwise, it becomes vulnerable to attackers. Thus, security services such as data confidentiality and authentication are required to secure the sensitive information. To provide such security services, networked devices must share a cryptographic key to secure the data. The current key distribution and establishment protocols, which are utilized for networked computers, are not applicable for resource-constrained devices due to their inherent characteristics (i.e., limited memory, computation, and processing powers). In this dissertation, we proposed lightweight key distribution schemes in three environments that encompass resource-constrained devices: Wireless Sensor Networks (WSNs), Fog Computing, and Blockchain Networks. Our lightweight cryptographic protocols require a small amount of memory and a smaller number of computations and low communication overhead compared to the other key distribution protocols.

In WSNs, we proposed, for the first time, two polynomial-based key pre-distribution

schemes with probabilistic security. The first scheme is designed as a hierarchical key distribution scheme in a hierarchical WSNs. With the hierarchical design, the pairwise keys are distributed in the network utilizing a top-down approach that proves to be an efficient and lightweight scheme for resource-constrained devices. The second proposed scheme is modeled as a non-interactive group key pre-distribution scheme in which a group key, also called a path key, is used to secure data transferred over the entire routing path. Specifically, instead of using link-to-link secure communication, which uses multiple pairwise shared keys, it uses an end-to-end secure communication that utilizes a single path key to protect data over the entire path. Although both schemes are lightweight and provide high connectivity in the WSNs, they suffer from sensor-captured attacks. To elaborate, the problem with all polynomial-based key distribution schemes is that the security of these schemes, which are called *deterministic k-secure*, depends on the degree of the chosen polynomial. In other words, if the degree of the chosen polynomial is k, then capturing the $k + 1$ sensors (or more) can compromise the systemâs security. Although increasing the degree of the polynomial can improve the security, it increases the storage and computational requirements of the sensors. In this dissertation, we propose, for the first time, polynomial-based key distribution schemes with probabilistic security. With the probabilistic security feature, we enhance the security of the WSNs against sensor-captured attacks. In addition, we show that the probability of a sensor capture attack can be significantly reduced by adjusting the system's parameters.

In fog computing, the environment in which the IoT devices represent end users' nodes, we proposed a hierarchical polynomial-based key management scheme. In our

proposed scheme, keys are distributed to the networked nodes in a top-down approach. The hierarchical structure of our proposed model significantly reduces the memory, processing, and communication requirements of the IoT's devices. In addition, our proposed scheme is scalable scheme in which more nodes can join the fog network without affecting the storage of other nodes that exist in the network.

In the blockchain network, we proposed a lightweight key management scheme in a permissioned blockchain environment utilizing bivariate polynomials. We noticed that most permissioned blockchains rely on a Public-Key Infrastructure (PKI) as cryptographic tools to provide security services such as identity authentication and data confidentiality. Using PKI to validate transactions includes validating digital certificates of endorsement peers that create an overhead in the system. Because public-key operations are computationally intensive, they limit the scalability of blockchain applications. Due to a large modulus size and expensive modular exponentiation operations, public-key operations such as RSA become slower than polynomial-based schemes that involve a smaller modulus size and an even smaller number of modular multiplications. For instance, the 2048-bit RSA is approximately 15,728 times slower than a polynomial with a degree of 50 and 128-bit modulus size. In this dissertation work, we propose a lightweight polynomial-based key management scheme in the context of a permissioned blockchain. Our scheme involves computationally less intensive polynomial evaluation operations such as additions and multiplications that result in a faster processing compared with public-key schemes. In addition, our proposed solution reduces the overhead of processing transactions and improves the system scalability. More importantly, the

proposed polynomial-based key distribution scheme is an unconditionally secure scheme, which is better than the security of the PKI schemes that are based on some computational assumptions. The one limitation of our proposed scheme is that it lacks a key revocation mechanism that we will consider in a future work.

7.1 Future Work

Our future direction is as follows:

1. Complete Lightweight Key Management Infrastructure

   According to NIST [105] *"The proper management of cryptographic keys is essential to the effective use of cryptography for security. Keys are analogous to the combination of a safe. If a safe combination is known to an adversary, the strongest safe provides no security against penetration. Similarly, poor key management may easily compromise strong algorithms."* Thus, when it comes to cryptographic keys, it is very important to think of the whole lifecycle of the keys from generating, updating, and deleting/revocation of keys. Ultimately, we aim at developing a complete lightweight key management scheme. We designed several key distribution/establishment schemes, and we plan to design a lightweight and efficient key revocation scheme. A Key Revocation Scheme is an important part of any key management scheme. When an attacker compromises a key or a key's lifetime expires, it is of great importance to have a mechanism to revoke the compromised keys and generate new keys and distribute them securely to entities in the network. For that reason, our goal is to design a key revocation scheme that is lightweight

for resource-constrained devices and efficient enough to enhance the security of the networked systems.

2. Applications for Lightweight Protocols

   Advancement in technology and wireless communications allows small devices such as cell phones and IoTs that are equipped with network connectivity to be part of any network infrastructure. Therefore, designing security protocols these days must consider these resource-limited devices that become an integral part of networked systems. In this dissertation, we consider three environments: Wireless Sensor Networks (WSN), Fog Computing, and Blockchain Networks. We plan to investigate other environments that involve resource-constrained devices such as *vehicular networks* and *smart homes*, and design lightweight cryptographic solutions for those environments.

3. New Models of Key Distributions

   Key distributions and establishment schemes aim at allowing two network nodes to share a key so they can communicate securely. In WSNs, great many approaches have been developed to allow any one pair of sensors to share a key that is used for secure communication. A random key distribution scheme, for example, is one of the most popular protocol in WSNs; it is a probabilistic key distribution that distributes keys between sensors based on some connectivity probability. If two strange sensors donât share a pairwise key, they start a *path key* discovery phase in which sensors search for a mutual neighboring node that shares pairwise keys with both sensors. That path discovery phase has high communication overhead.

The aforementioned scheme is not a centralized key establishment, but the key establishment task is distributed between sensors, which is a complicated task for such resource-constrained devices. For that reason, we plan to develop a scheme that facilitates key establishment between two strange sensors in WSNs based on pre-shared pairwise key schemes (i.e., random key scheme). In contrast to the path discovery phase in which the mutual neighbor generates a key and distributes it securely to the two sensors, our approach allows the mutual neighbor to utilize the pre-shared pairwise keys with the two sensors to generate a shared key using lightweight techniques such as broadcasting and an XOR operation. That is, our goal is to not only design a lightweight scheme but to increase the probability of key establishment by designing a new model that is built on top of the existing pre-shared pairwise key schemes.

4. Attack Detection Schemes

   We plan to develop an attack detection protocol based on a cryptographic puzzle approach and machine-learning techniques. Currently, we are working on a cryptographic scheme that aims at detecting Sybil attacks in distributed systems. In Sybil attack, an adversary controls several identities in a way that can manipulate the network. Such attacks are popular in WSNs and also other distributed systems such as blockchain networks. With machine-learning techniques, we can build a protocol that can detect attacks based on behavioral attributes [106].

5. Testing Platform

   We aim at building a real testing platform such as raspberry-pi clusters to test the

real performance of our schemes and compare them with other practical protocols. The testing platform is very important to evaluate the performance of our protocols, to test the practical part of our approaches, and to get practical data that could be utilized for data analysis. With the testing platform we can simulate different real-world scenarios.

REFERENCE LIST

[1] D. Kim and M. G. Solomon, *Fundamentals of Information Systems Security*, 3rd ed. Burlington, MA: Jones & Bartlett Learning, Oct. 2016.

[2] Merriam-Webster, "Definition of SECURITY." [Online]. Available: https://www.merriam-webster.com/dictionary/security

[3] M. Bishop, *Introduction to Computer Security*, 1st ed. Boston, MA: Addison-Wesley Professional, Nov. 2004.

[4] D. Morgan, D. Taylor, and G. Custeau, "A Survey of Methods for Improving Computer Network Reliability and Availability," *Computer*, vol. 10, no. 11, pp. 42–50, Nov. 1977. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1646296

[5] N. Ferguson, B. Schneier, and T. Kohno, *Cryptography Engineering: Design Principles and Practical Applications*, 1st ed. Indianapolis, IN: Wiley, Mar. 2010.

[6] D. R. Stinson, *Cryptography: Theory and Practice*, 3rd ed. Boca Raton, FL: Chapman and Hall/CRC, Nov. 2005.

[7] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-Secure Key Distribution for Dynamic Conferences," in *Advances in Cryptology â CRYPTO' 92*, E. F. Brickell, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 471–486.

[8] K. Sohraby, D. Minoli, and T. Znati, *Wireless Sensor Networks: Technology, Protocols, and Applications*. New York, NY: Wiley-Interscience, 2007.

[9] D. Evans, "How the Next Evolution of the Internet Is Changing Everything," p. 11, 2011. [Online]. Available: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[10] M. Abdelshkour, "IoT, from Cloud to Fog Computing," Mar. 2015. [Online]. Available: https://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing

[11] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," in *2012 Mobile Cloud Computing Conf.* Helsinki, Finland: ACM, 2012, pp. 13–16. [Online]. Available: http://doi.acm.org/10.1145/2342509.2342513

[12] D. Drescher, *Blockchain Basics: A Non-Technical Introduction in 25 Steps*, 1st ed. Berkeley, California: Apress, Mar. 2017.

[13] H. Wang, Z. Zheng, S. Xie, H. N. Dai, and X. Chen, "Blockchain challenges and opportunities: a survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018. [Online]. Available: http://www.inderscience.com/link.php?id=10016848

[14] R. Guo, H. Shi, Q. Zhao, and D. Zheng, "Secure Attribute-Based Signature Scheme With Multiple Authorities for Blockchain in Electronic Health Records Systems," *IEEE Access*, vol. 6, pp. 11 676–11 686, 2018.

[15] L. Ao, C. Ogah, P. Asuquo, H. Cruickshank, and S. Zhili, "A Secure Key Management Scheme for Heterogeneous Secure Vehicular Communication Systems," *ZTE Commun*, vol. 14, no. S0, pp. 21–31, 2016.

[16] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in *2017 IEEE Int. Conf. on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Mar. 2017, pp. 618–623.

[17] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. Boston: Pearson, Mar. 2016.

[18] W. Stallings and L. Brown, *Computer Security: Principles and Practice*, 3rd ed. Boston: Pearson, Jul. 2014.

[19] W. Fumy, "Key management techniques," in *State of the Art in Applied Cryptography, Course on Computer Security and Industrial Cryptography - Revised Lectures*. London, UK, UK: Springer-Verlag, 1998, pp. 142–162. [Online]. Available: http://dl.acm.org/citation.cfm?id=647443.726910

[20] M. A. Simplício, P. S. L. M. Barreto, C. B. Margi, and T. C. M. B. Carvalho, "A survey on key management mechanisms for distributed Wireless Sensor Networks," *Computer Networks*, vol. 54, no. 15, pp. 2591–2612, Oct. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S138912861000112X

[21] R. Blom, "Non-Public Key Distribution," in *Advances in Cryptology*, D. Chaum, R. L. Rivest, and A. T. Sherman, Eds. Boston, MA: Springer US, 1983, pp. 231–236.

[22] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Commun. ACM*, vol. 21, no. 12, pp. 993–999, Dec. 1978. [Online]. Available: http://doi.acm.org/10.1145/359657.359659

[23] J. Kohl and C. Neuman, "The Kerberos Network Authentication Service (V5)," 1993. [Online]. Available: https://tools.ietf.org/html/rfc1510

[24] W. Diffie and M. E. Hellman, "Multiuser cryptographic techniques," in *Proceed. of the National Computer Conf. and Exposition*, ser. AFIPS '76. New York, NY, USA: ACM, 1976, pp. 109–112. [Online]. Available: http://doi.acm.org/10.1145/1499799.1499815

[25] R. Blom, "An optimal class of symmetric key generation systems," in *Proc. Of the EUROCRYPT 84 Workshop on Advances in Cryptology: Theory and Application of Cryptographic Techniques*. New York, NY, USA: Springer-Verlag New York, Inc., 1985, pp. 335–338. [Online]. Available: http://dl.acm.org/citation.cfm?id=20177.20199

[26] D. Liu, P. Ning, and W. Du, "Group-based key pre-distribution in wireless sensor networks," in *Proceedings of the 4th ACM Workshop on Wireless Security*, ser. WiSe '05. New York, NY, USA: ACM, 2005, pp. 11–20. [Online]. Available: http://doi.acm.org/10.1145/1080793.1080798

[27] X. Fan and G. Gong, "LPKM: A Lightweight Polynomial-Based Key Management Protocol for Distributed Wireless Sensor Networks," in *Ad Hoc Networks*, J. Zheng, N. Mitton, J. Li, and P. Lorenz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 180–195.

[28] Y. Liu and Y. Wu, "A Key Pre-distribution Scheme based on Sub-regions for Multi-Hop Wireless Sensor Networks," *Wireless Personal Communications*, vol. 109, no. 2, pp. 1161–1180, Nov. 2019. [Online]. Available: https://doi.org/10.1007/s11277-019-06608-3

[29] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, ser. CCS '02. New York, NY, USA: ACM, 2002, pp. 41–47. [Online]. Available: http://doi.acm.org/10.1145/586110.586117

[30] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, ser. SP '03. IEEE Computer Society, 2003, pp. 197–. [Online]. Available: http://dl.acm.org/citation.cfm?id=829515.830566

[31] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *IEEE INFOCOM 2004*, vol. 1, 2004, p. 597.

[32] A. Rasheed and R. Mahapatra, "Key Predistribution Schemes for Establishing Pairwise Keys with a Mobile Sink in Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 1, pp. 176–184, 2011.

[33] S. Ruj, A. Nayak, and I. Stojmenovic, "Pairwise and Triple Key Distribution in Wireless Sensor Networks with Applications," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2224–2237, 2013.

[34] O. Yagan and A. M. Makowski, "Wireless Sensor Networks Under the Random Pairwise Key Predistribution Scheme: Can Resiliency Be Achieved With Small Key Rings?" *IEEE/ACM Trans. Netw.*, vol. 24, no. 6, pp. 3383–3396, 2016. [Online]. Available: https://doi.org/10.1109/TNET.2016.2527742

[35] J. Ding, A. Bouabdallah, and V. Tarokh, "Key Pre-Distributions From Graph-Based Block Designs," *IEEE Sensors Journal*, vol. 16, no. 6, pp. 1842–1850, 2016.

[36] F. Gandino, R. Ferrero, and M. Rebaudengo, "A Key Distribution Scheme for Mobile Wireless Sensor Networks: q-s -Composite," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 34–47, 2017.

[37] L. Harn, C. Hsu, O. Ruan, and M. Zhang, "Novel design of secure end-to-end routing protocol in wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 6, pp. 1779–1785, March 2016.

[38] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications*

Security, ser. CCS '03. New York, NY, USA: ACM, 2003, pp. 52–61. [Online]. Available: http://doi.acm.org/10.1145/948109.948119

[39] Y. Cheng and D. P. Agrawal, "An improved key distribution mechanism for large-scale hierarchical wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 35 – 48, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S157087050600045X

[40] S. Zhao, K. E. Tepe, I. Seskar, and D. Raychaudhuri, "Routing protocols for self-organizing hierarchical ad-hoc wireless networks," in *IEEE Sarnoff Symposium*, 2003. [Online]. Available: http://www.winlab.rutgers.edu/~sulizhao/isRtHierAdhoc.pdf

[41] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, March 2000.

[42] B. Liu, Z. Liu, and D. Towsley, "On the capacity of hybrid wireless networks," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, vol. 2, March 2003, pp. 1543–1552 vol.2.

[43] An-Ni Shen, Song Guo, and Hung-Yu Chien, "An efficient and scalable key distribution mechanism for hierarchical wireless sensor networks," in *2009 IEEE Sarnoff Symposium*. Piscataway, NJ, USA: IEEE Press, March 2009, pp. 1–5.

[44] K. N. Kumar and M. J. Nene, "Chip-based symmetric and asymmetric key generation in hierarchical wireless sensors networks," in *2017 International Conference on Inventive Systems and Control (ICISC)*. IEEE, Jan 2017, pp. 1–6.

[45] I. Akyildiz and M. C. Vuran, *Wireless Sensor Networks*. New York, NY, USA: John Wiley & Sons, Inc., 2010.

[46] Z. Mahmood, H. Ning, and A. Ghafoor, "A Polynomial Subset-Based Efficient Multi-Party Key Management System for Lightweight Device Networks," *Sensors*, vol. 17, no. 4, p. 670, Mar. 2017. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/28338632

[47] A. G. Dinker and V. Sharma, "Trivariate Polynomial Based Key Management Scheme (TPB-KMS) in Hierarchical Wireless Sensor Networks," in *Ambient Communications and Computer Systems.Advances in Intelligent Systems and Computing*, G. M. Perez, S. Tiwari, M. C. Trivedi, and K. K. Mishra, Eds. Singapore: Springer Singapore, 2018, pp. 283–290.

[48] P. N. Bahrami, H. H. Javadi, T. Dargahi, A. Dehghantanha, and K.-K. R. Choo, "A hierarchical key pre-distribution scheme for fog networks," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 22, p. e4776, 2019, e4776 cpe.4776. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4776

[49] A. Albakri, M. Maddumala, and L. Harn, "Hierarchical polynomial-based key management scheme in fog computing," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Aug 2018, pp. 1593–1597.

[50] K. Hamsha and G. S. Nagaraja, "Threshold Cryptography Based Light Weight Key Management Technique for Hierarchical WSNs," in *Ubiquitous Communications and Network Computing (UBICNET 2019).Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, N. Kumar and R. Venkatesha Prasad, Eds., vol. 276. Cham: Springer International Publishing, 2019, pp. 188–197.

[51] A. Kumar, N. Bansal, and A. R. Pais, "New key pre-distribution scheme based on combinatorial design for wireless sensor networks," *IET Communications*, vol. 13, no. 7, pp. 892–897, 2019.

[52] E. Khan, E. Gabidulin, B. Honary, and H. Ahmed, "Matrix-based memory efficient symmetric key generation and pre-distribution scheme for wireless sensor networks," *IET Wireless Sensor Systems*, vol. 2, no. 2, pp. 108–114, 2012. [Online]. Available: http://digital-library.theiet.org/content/journals/10.1049/iet-wss.2011.0097

[53] J.-P. Sheu and J.-C. Cheng, "Pair-wise path key establishment in wireless

sensor networks," *Computer Communications*, vol. 30, no. 11, pp. 2365–2374, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366407001764

[54] L. Harn and G. Gong, "Conference key establishment protocol using a multivariate polynomial and its applications," *Security and Communication Networks*, vol. 8, no. 9, pp. 1794–1800, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.1143

[55] L. Harn and C. Hsu, "Predistribution Scheme for Establishing Group Keys in Wireless Sensor Networks," *IEEE Sensors Journal*, vol. 15, no. 9, pp. 5103–5108, 2015.

[56] M. Ge, K.-K. R. Choo, H. Wu, and Y. Yu, "Survey on key revocation mechanisms in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 63, no. C, pp. 24–38, Mar. 2016. [Online]. Available: https://doi.org/10.1016/j.jnca.2016.01.012

[57] D. Mall, K. Konatè, and A. K. Pathan, "On the key revocation schemes in wireless sensor networks," in *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, Aug 2013, pp. 290–297.

[58] B. Z. Abbasi and M. A. Shah, "Fog computing: Security issues, solutions and robust practices," in *2017 23rd International Conference on Automation and Computing (ICAC)*. IEEE, Sep. 2017, pp. 1–6.

[59] K. Lee, D. Kim, D. Ha, U. Rajput, and H. Oh, "On security and privacy issues of fog computing supported internet of things environment," in *2015 6th International Conference on the Network of the Future (NOF)*, Sep. 2015, pp. 1–3.

[60] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced iot," *IEEE Access*, vol. 5, pp. 3302–3312, mar 2017.

[61] A. B. Amor, M. Abid, and A. Meddeb, "A privacy-preserving authentication scheme in an edge-fog environment," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, Oct 2017, pp. 1225–1231.

[62] A. Alrawais, A. Alhothaily, C. Hu, X. Xing, and X. Cheng, "An attribute-based encryption scheme to secure fog communications," *IEEE Access*, vol. 5, pp. 9131–9138, may 2017.

[63] P. Porambage, A. Braeken, A. Gurtov, M. Ylianttila, and S. Spinsante, "Secure end-to-end communication for constrained devices in iot-enabled ambient assisted living systems," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, Dec 2015, pp. 711–714.

[64] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theor.*, vol. 22, no. 6, pp. 644–654, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/TIT.1976.1055638

[65] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979. [Online]. Available: http://doi.acm.org/10.1145/359168.359176

[66] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An ID-Based Linearly Homomorphic Signature Scheme and Its Application in Blockchain," *IEEE Access*, vol. 6, pp. 20 632–20 640, 2018.

[67] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: Challenges and Solutions," 2016. [Online]. Available: http://arxiv.org/abs/1608.05187

[68] T. Gao, D. Greenspan, M. Welsh, R. R. Juang, and A. Alm, "Vital signs monitoring and patient tracking over a wireless network," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. IEEE, Jan 2005, pp. 102–105.

[69] O. Cheikhrouhou, "Secure group communication in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 61, no. C, pp. 115–132, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2015.10.011

[70] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Comm. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002. [Online]. Available: https://doi.org/10.1109/MCOM.2002.1024422

[71] S. Hu, "A hierarchical key management scheme for wireless sensor networks based on identity-based encryption," in *2015 IEEE International Conference on Computer and Communications (ICCC)*. IEEE, Oct 2015, pp. 384–389.

[72] M. Alshammari and K. Elleithy, "Secure and efficient key management protocol (sekmp) for wireless sensor networks," in *Proceedings of the Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, ser. ANCS '14. New York, NY, USA: ACM, 2014, pp. 253–254. [Online]. Available: http://doi.acm.org/10.1145/2658260.2661775

[73] N. Saqib and U. Iqbal, "Security in wireless sensor networks using ECC," in *2016 IEEE International Conference on Advances in Computer Applications (ICACA)*. IEEE, Oct 2016, pp. 270–274.

[74] A. Albakri, L. Harn, and S. Song, "Hierarchical Key Management Scheme with Probabilistic Security in a Wireless Sensor Network (WSN)," *Security and Communication Networks*, vol. 2019, 2019. [Online]. Available: https://www.hindawi.com/journals/scn/2019/3950129/cta/

[75] M. P. Durisić, Z. Tafa, G. Dimić, and V. Milutinović, "A survey of military applications of wireless sensor networks," in *2012 Mediterranean Conference on Embedded Computing (MECO)*. IEEE, June 2012, pp. 196–199.

[76] T. Azzabi, H. Farhat, and N. Sahli, "A survey on wireless sensor networks security issues and military specificities," in *2017 International Conference on Advanced Systems and Electric Technologies (IC$_A$SET)*. $IEEE, Jan2017, pp.66 - -72.$

[77] I. Sinclair, *Sensors and Transducers*, 3rd ed. Woburn, MA: Butterworth-Heinemann, 2001.

[78] S. Zhu, S. Setia, and S. Jajodia, "Leap+: Efficient security mechanisms for large-scale distributed sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 4, pp. 500–528, 11 2006.

[79] B. Maala, H. Bettahar, and A. Bouabdallah, "TLA: A tow level architecture for key management in wireless sensor networks," in *2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*. IEEE, Aug 2008, pp. 639–644.

[80] X. Zhang and J. Wang, "An efficient key management scheme in hierarchical wireless sensor networks," in *2015 International Conference on Computing, Communication and Security (ICCCS)*. IEEE, Dec 2015, pp. 1–7.

[81] L. Harn, C.-F. Hsu, Z. Xia, and J. Zhou, "How to share secret efficiently over networks," *Security and Communication Networks*, vol. 2017, 2017. [Online]. Available: https://www.hindawi.com/journals/scn/2017/5437403/

[82] Y. Wang, B. Ramamurthy, and X. Zou, "Keyrev: An efficient key revocation scheme for wireless sensor networks," in *2007 IEEE International Conference on Communications*. IEEE, June 2007, pp. 1260–1265.

[83] P. T. C., G. B. Kianoosh, M. H. Amini, N. Sunitha, and S. Iyengar, "Key pre-distribution scheme with join leave support for scada systems," *International Journal of Critical Infrastructure Protection*, vol. 24, pp. 111 – 125, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1874548217301038

[84] D. E. Knuth, *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*, 3rd ed. Addison-Wesley Longman Publishing Co., Inc., 1997.

[85] A. Albakri and L. Harn, "Non-interactive group key pre-distribution scheme (gkps) for end-to-end routing in wireless sensor networks," *IEEE Access*, vol. 7, pp. 31 615–31 623, 2019.

[86] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978. [Online]. Available: http://doi.acm.org/10.1145/359340.359342

[87] M. Ketel, "Fog-cloud services for iot," in *Proceedings of the SouthEast Conference*, ser. ACM SE â17. New York, NY, USA: Association for Computing Machinery, 2017, p. 262â264. [Online]. Available: https://doi.org/10.1145/3077286.3077314

[88] R. Mahmud, R. Kotagiri, and R. Buyya, *Fog Computing: A Taxonomy, Survey and Future Directions*. Singapore: Springer Singapore, 2018, pp. 103–130. [Online]. Available: https://doi.org/10.1007/978-981-10-5861-5_5

[89] Y. Guan, J. Shao, G. Wei, and M. Xie, "Data security and privacy in fog computing," *IEEE Network*, vol. 32, no. 5, pp. 106 –111, Sep. 2018.

[90] M. Mukherjee, R. Matam, L. Shu, L. Maglaras, M. A. Ferrag, N. Choudhury, and V. Kumar, "Security and privacy in fog computing: Challenges," *IEEE Access*, vol. 5, pp. 19 293–19 304, 2017.

[91] J. Zhang and V. Varadharajan, "Wireless sensor network key management survey and taxonomy," *J. Netw. Comput. Appl.*, vol. 33, no. 2, p. 63â75, Mar. 2010. [Online]. Available: https://doi.org/10.1016/j.jnca.2009.10.001

[92] A. Panarello, N. Tapas, G. Merlino, F. Longo, A. Puliafito, A. Panarello, N. Tapas, G. Merlino, F. Longo, and A. Puliafito, "Blockchain and IoT Integration: A Systematic Survey," *Sensors*, vol. 18, no. 8, p. 2575, Aug. 2018.

[93] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain Challenges and Opportunities: A Survey," *Inderscience Publishers*, p. 25, 2017.

[94] H. T. Vo, A. Kundu, and M. Mohania, "Research Directions in Blockchain Data Management and Analytics," 2018.

[95] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, May 2018.

[96] M. Vukolić, "Rethinking Permissioned Blockchains," in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts - BCC '17*. Abu Dhabi, United Arab Emirates: ACM Press, 2017, pp. 3–7.

[97] E. Androulaki, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, A. Barger, S. W. Cocco, J. Yellick, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, and G. Laventman, "Hyperledger fabric: A distributed operating system

for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference on - EuroSys '18*. Porto, Portugal: ACM Press, 2018, pp. 1–15.

[98] C. Cachin, "Architecture of the Hyperledger Blockchain Fabric," *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, p. 4, Jul. 2016, chicago, Illinois, USA.

[99] "Peers — hyperledger-fabricdocs master documentation," https://hyperledger-fabric.readthedocs.io/en/release-1.2/peers/peers.html.

[100] "Transaction Flow — hyperledger-fabricdocs master documentation," https://hyperledger-fabric.readthedocs.io/en/release-1.2/txflow.html.

[101] "Channels — hyperledger-fabricdocs master documentation," https://hyperledger-fabric.readthedocs.io/en/release-1.2/channels.html.

[102] "Private data — hyperledger-fabricdocs master documentation," https://hyperledger-fabric.readthedocs.io/en/release-1.2/private-data/private-data.html.

[103] A. Pellegrini, V. Bertacco, and T. Austin, "Fault-based attack of RSA authentication," in *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, Mar. 2010, pp. 855–860.

[104] "Fabric CA User's Guide — hyperledger-fabric-cadocs master documentation," https://hyperledger-fabric-ca.readthedocs.io/en/latest/users-guide.html/fabric-ca-server.

[105] E. Barker, “Recommendation for Key Management Part 1: General,” National Institute of Standards and Technology, Tech. Rep. NIST SP 800-57pt1r4, Jan. 2016. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf

[106] K. N. Junejo and J. Goh, “Behaviour-based attack detection and classification in cyber physical systems using machine learning,” in *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*, ser. CPSS â16. New York, NY, USA: Association for Computing Machinery, 2016, p. 34â43. [Online]. Available: https://doi.org/10.1145/2899015.2899016

VITA

Ashwag Albakri was born on August 11, 1987 in Lincoln Shire, UK.

Ashwag Albakri received her B.S. degree in Computer Science from King Abdulaziz University, Saudi Arabia in 2010., She obtained her M.S. degree in Information Science from the University of Pittsburgh, PA in 2015 where she specialized in Security Assured Information Systems. She is a lecturer in the Computer Science department at Jazan University in Saudi Arabia. During her PhD, Ms, Albakri worked as a Graduate Teaching Assistance (GTA) and taught *an Introduction to Cryptology* course. Her research interests are cryptography, information security, and network security. She has four publications: two journal papers, which were published in 2019 in the *IEEE Access* and *Security and Communication Networks* Journal, and two conference papers were published in the *IEEE International Conference on Trust, Security, and Privacy in Computing and Communications* (TrustCom) in 2018 , and in the *IEEE Conference on Communications and Network Security* (CNS) in 2019. She also presented a poster at the 40th *IEEE Security and Privacy Conference* in San Francisco, CA.

In addition, Ms. Albakri was a visiting researcher at the University of Southern California where she worked on developing a sybil attack detection mechanism.

Moreover, Ashwag received several awards: The Graduate Assistance Fund (GAF) awards in 2019, the Grace Hopper Celebration Student Scholarship in 2018, and several distinguished awards from the Saudi Arabia Cultural Mission in 2015. Ashwag has several professional certificates: Certified Blockchain Expert (CBE), Information System

Security Professionals (CNSS 4011), Designated Approving Authority (4012), System Administrators in Information Systems Security (4013), and a Preparing Future Faculty certificate.