

DECENTRALIZED KEY DISTRIBUTION ALGORITHM FOR IoT DEVICES

PHASE II REPORT

Submitted by

K GOKULAKRISHNAN

K DHANUSH

VP PRANAV VISHAL

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

ELECTRONICS & COMMUNICATION ENGINEERING



**Department of Electronics & Communication
Engineering**

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

MAY 2024

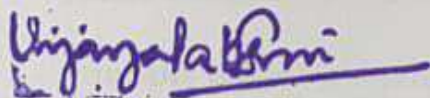
Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this Report titled “**DECENTRALISED KEY DISTRIBUTION ALGORITHM FOR IoT DEVICES**” is the bonafide work of **K GOKULAKRISHNAN (203002031), K DHANUSH (203002026) and VP PRANAV VISHAL (203002081)** who carried out the work under my supervision.

Certified further that to the best of my knowledge the work reported here does not form part of any other thesis or dissertation because of which a degree or award was conferred on an earlier occasion on this or any other candidate.



Dr. P. Vijayalakshmi

HEAD OF THE DEPARTMENT

Professor & Head of the Department
Electronics & Communication Engineering
SSN College of Engineering
Kalavakkam – 603 110



Dr. A. Jawahar

SUPERVISOR

Professor
Electronics & Communication Engineering
SSN College of Engineering
Kalavakkam – 603 110

Submitted for Project Viva-Voce Examination held on.....

EXTERNAL EXAMINER

INTERNAL EXAMINER

ABSTRACT

The widespread adoption of Internet of Things (IoT) devices handling sensitive data has prompted the development of advanced access control technologies to safeguard this information from unauthorized access. In dynamic IoT environments, characterized by frequent subscriber mobility and high signaling overhead, ensuring secure data distribution to legitimate subscribers becomes paramount. Group Key Management (GKM) emerges as a crucial mechanism for managing key dissemination for access control and secure data distribution in such dynamic settings. However, existing access control schemes based on GKM, specifically tailored for IoT, primarily rely on centralized models. These models face challenges in addressing the scalability issue posed by the sheer volume of IoT devices and the growing number of subscribers. Additionally, current GKM schemes lack support for member independence within the same group, focusing instead on dependent symmetric group keys for subgroup communication, which proves inefficient for highly dynamic subscriber behavior. To tackle these challenges, we propose a novel architecture called Decentralized Lightweight Group Key Management for Access Control in the IoT environment (DLGKMAC). This architecture, based on a hierarchical structure

comprising a Key Distribution Center (KDC) and several Sub Key Distribution Centers (SKDCs), enhances subscriber group management and reduces rekeying overhead on the KDC. Furthermore, our approach introduces a new master token management protocol to oversee key dissemination across subscriber groups, minimizing storage, computation, and communication overheads during join/leave events. DLGKM-AC accommodates a scalable IoT architecture, mitigating single points of failure by distributing the rekeying load across the network. It ensures secure group communication by thwarting collusion attacks and ensuring backward/forward secrecy. In our implementation, we develop a graphical user interface (GUI) for RSA algorithm.

ACKNOWLEDGEMENT

We express our heartfelt gratitude to **Dr. V. E. Annamalai**, Principal, SSN College of Engineering, for the facilities provided to us during the project. We take this opportunity to express our profound gratitude to **Dr. P. Vijayalakshmi**, HOD of the ECE Department, for her exemplary guidance, monitoring, and constant encouragement throughout the course of this project. We also take this opportunity to express a deep gratitude to **Dr. A. Jawahar**, Professor at SSN College of Engineering, for their guidance and support which helped us in completing tasks. We are also obliged to the panel members **Dr. R. Hemalatha**, Associate Professor at SSN College of Engineering and **Dr. S. Aasha Nandhini**, Assistant Professor at SSN College of Engineering, for their support. We take this opportunity to thank all the staff members of the Department of Electronics and Communication Engineering, our family and friends for their constant support at all times.

TABLE OF CONTENTS

CHAPTER No	TITLE	PAGE No
	ABSTRACT	i
	ACKNOWLEDGEMENT	iii
	LIST OF TABLES	vii
	LIST OF FIGURES	viii
	LIST OF SYMBOLS AND ABBREVIATIONS	x
1	INTRODUCTION	1
	1.1 Overview	1
	1.2 Motivation	3
	1.3 Objective of the Study	3
2	LITERATURE SURVEY	5
3	PROPOSED WORK	9
		10
	3.2 Examine Attack Scenarios	11
		11
4	PROPOSED MODEL	13
	4.1 Key Generation Algorithm	15

	4.2 Master Key Encryption	17
	4.3 Scheme Description	18
		19
5	SIMULATION RESULTS	
	5.1 Introduction	
	5.2 Initial Interface	
	5.3 Registration of Users	
	5.4 Login Page For Users	
	5.5 Encryption of Text Files	
	5.6 Decryption of Text Files	
	5.7 Downloading Decrypted Files	
	5.8 Analysis of Keygen Algorithm	
	5.9 Sequencing to Avoid Replay Attacks	
6	ANALYSIS AND EVALUATION	
	6.1 Security Analysis	
	6.1.1 Forward Security	
	6.1.2 Backward Security	
	6.2 Performance Analysis	
	6.2.1 Storage Overhead	

		vi
	6.2.2 Computation Overhead	40
	6.2.2.1 When a User Joins a User Group	40
	6.2.2.2 When a User Leaves a User Group	41
	6.2.2.3 When a Device Joins a User Group	41
	6.2.2.4 When a Device Leaves a Device Group	41
	6.2.3 Communication Overhead	42
	6.2.4 Performance Evaluation	43
7	CONCLUSION	47
	REFERENCES	49

LIST OF TABLES

TABLE No	TITLE	PAGE No
5.1	Excel Sheet with New Users Details	26
5.2	Excel Sheet with New Registered Users Details	26
6.1	Communication Overhead	43

LIST OF FIGURES

FIGURE No	TITLE	PAGE No
3.1	Methodology of Analysing The DLGKM-AC Algorithm	10
4.1	Proposed System Model	14
4.2		16
5.1	GUI to Register Users and Login	25
5.2	GUI to Register User	26
5.3	GUI to Login User	27
5.4	User Not Found Pop up	28
5.5	Invalid Device Group Pop up	28
5.6	Invalid Password Pop up	28
5.7	Login Successful Pop up	28
5.8	GUI of Main Window	
5.9	GUI Window after encrypting the required text file	

5.10	Pop up asking user's private key	31
5.11	Pop up asking device group's public key	31
5.12	GUI Window after Decrypting the the required text file	31
5.13	Browsing Window after Triggering the Download Button	32
5.14	GUI Window after Downloading the Decrypted text file	33
5.15	Plain Text File Downloaded	33
5.16	Analysis Of KeyGen Algorithm	34
5.17	Sequence Numbering	35
6.1	Computation Overhead	45
6.2	Computation Overhead per Group	46

LIST OF ABBREVIATIONS

ABBREVIATION	ANNOTATION
CRT	Chinese Remainder Theorem
DG	Device Group
DGKM - AC	Decentralized Group Key Management for Access Control
GCRT	Group Chinese Remainder Theorem
GKM	Group Key Management
KDC	Key Distribution Center
KEK	Key Encryption Key
LKH	Logical Key Hierarchy
MKeyGen	Master Key Generation
OFT	One-way Function Tree
SKDC	Sub Key Distribution Center
TEK	Traffic Encryption Key
UG	User Group

CHAPTER 1 - INTRODUCTION

1.1. OVERVIEW

The Internet of Things (IoT) has emerged as a pervasive paradigm connecting a multitude of digital devices to the Internet seamlessly. In recent times, IoT devices have become increasingly integrated into people's daily lives, forming a diverse network of interconnected physical objects with various functionalities, with data collection being a key feature. However, most IoT devices possess minimal computational, communication, and storage resources, which hinders their ability to efficiently perform cryptographic operations, thereby presenting significant security challenges. Despite its widespread adoption, large-scale IoT deployments still grapple with serious security concerns, including authentication, privacy preservation, and data integrity.

Given the critical need to protect IoT data from tampering and unauthorized access, implementing an effective access control scheme is imperative. Group Key Management (GKM) emerges as a promising approach to provide access control to data streams exclusively for legitimate users.

1.2. MOTIVATION

Secure communication plays a vital role in maintaining the integrity and confidentiality of data exchanged among IoT devices. The heightened vulnerability of IoT devices to cyberattacks due to their often resource-constrained nature should be reduced to ensure secured communication between several devices in a system. The traditional method of centralized key management has advantages like single point failure, security risk, and scalability challenges that can be efficiently replaced by decentralized key algorithm.

To manage cryptographic key distribution securely and efficiently within IoT networks. Securing sensitive data involves preventing unauthorized access and safeguarding against a range of attacks, including eavesdropping, data tampering, impersonation, and replay attacks. To ensure that the algorithm is scalable to handle the dynamic addition and removal of devices without significantly impacting performance or security.

1.3. OBJECTIVE OF THE STUDY

However, existing GKM schemes often prove unsuitable for IoT applications. Typically, these schemes are tailored to manage communication within a single group, and those designed for access control within the IoT environment predominantly rely on

centralized architectures. Consequently, they fail to accommodate the scalability and dynamism inherent in IoT environments comprising multiple groups. Furthermore, with users subscribing to numerous services from various IoT devices and frequently changing their interests, maintaining efficient GKM in such dynamic environments poses a significant challenge, primarily due to the rekeying process affecting all group members during joining or leaving events [1].

To address these challenges, this research introduces a novel Decentralized Lightweight Group Key Management Architecture for Access Control, termed DLGKM-AC. This architecture aims to mitigate the dependence on rekeying, minimize resulting overhead, and achieve scalable access management in dynamic IoT environments. By adopting a decentralized approach, DLGKM-AC seeks to enhance the security and efficiency of access control mechanisms in IoT ecosystems [2].

The main idea of DGKM-AC is to create an efficient and flexible mechanism to secure distribution of contents to eligible subscribers. In this project, a hierarchical architecture is adopted using one Key Distribution Center (KDC), for managing group keys and broadcasting update messages, and several Sub – Key

Distribution Centers (SKDCs), for handling direct communication links between devices and users. A new master token encryption algorithm is used to ensure member's independence.

CHAPTER 2 - LITERATURE SURVEY

In the realm of Internet of Things (IoT), the secure distribution of cryptographic keys among decentralized devices stands as a critical challenge. As the IoT ecosystem burgeons, encompassing diverse applications from smart homes to industrial automation, ensuring robust security measures becomes paramount. Decentralized key distribution algorithms emerge as pivotal solutions to address the vulnerabilities inherent in centralized systems, offering enhanced resilience and scalability. This literature review navigates through the landscape of decentralized key distribution algorithms tailored for IoT environments, probing into their theoretical underpinnings, practical implementations, and efficacy in mitigating security threats. By synthesizing existing research, this review endeavors to elucidate the evolving paradigms, unresolved challenges, and future directions in the domain, thereby contributing to the ongoing discourse on securing IoT ecosystems.

Maissa Dammak et al. (2020) [3] Users can always get access to data even if one SKDC is affected. Extensive security analysis covering a wide range of desired security properties has also been provided. Group key management with logic key hierarchy and master key encryption.

Joseph N. Mamvong et al. (2021) [4] Secure authentication using the ATECC608A. An efficient security algorithm for power constrained IoT devices that aimed to reduce the complexity of the currently used security algorithm.

Sarika Y. Bonde et al. (2017) [5] Performance evaluation is done using RSA and SRNN algorithms. For further evaluation a 2 key pair algorithm is used. In communication security encryption algorithms play an essential role where encryption time is the major issue of concern. For performance evaluation RSA algorithm, 2 key pair algorithm and short-range natural number (SRNN) algorithm are used.

K. Sundarakantham et al. (2022) [6] The BMU-IOT method is used for the formation of subgroups from the cluster head. The SCK is generated using a unicast method. An efficient security algorithm for power constrained IoT devices that aimed to reduce the complexity of the currently used security algorithm.

Wenjuan Zhang et al. (2018) [7] GKM with DLGKM AC method and the attacks which are vulnerable to it. Group key management using DLGKM-AC and the threat models and the attacks related to this algorithm.

In the ever-expanding landscape of the Internet of Things (IoT), security concerns continue to evolve, and one particularly challenging aspect is the vulnerability to replay attacks. Despite the multitude of cryptographic protocols designed to enhance the security of IoT devices, there exists a notable research gap in formulating solutions that effectively prevent replay attacks while concurrently preserving optimal performance parameters. The need for lightweight and efficient security protocols is pronounced, especially in scenarios where the devices operate on constrained computational resources, low power, and limited bandwidth.

The modified DLGKM-AC algorithm proposed in this research aims to fill this research gap by introducing a novel approach that centers on sequencing to mitigate replay attacks. The significance of this lies in its potential to provide a robust defense against replay attacks, a known vulnerability in IoT communication, without unduly burdening the performance metrics critical for the seamless operation of these devices.

By addressing the specific challenges associated with replay attacks in the IoT context, the proposed modification to the DLGKMAC algorithm seeks to contribute to the development of more resilient and efficient communication protocols for IoT

devices. This research endeavors to provide a nuanced solution that not only bolsters security but does so in a manner mindful of the limitations inherent in IoT devices, thereby offering a promising avenue for advancing the state-of the-art in secure and performance of IoT communication.

CHAPTER 3 - PROPOSED WORK

In the burgeoning landscape of the Internet of Things (IoT), the secure distribution of cryptographic keys among decentralized devices remains a pivotal concern, demanding innovative solutions to fortify the IoT ecosystem against evolving security threats. This proposed work aims to explore and develop a novel decentralized key distribution algorithm specifically tailored for IoT devices. With the proliferation of interconnected devices across various domains, ranging from smart homes to industrial automation, the need for robust security measures has become increasingly pronounced. By leveraging decentralized architectures, the proposed algorithm endeavors to enhance the resilience, scalability, and cryptographic robustness of key distribution mechanisms within IoT environments. Through this endeavor, the proposed work seeks to contribute to the advancement of secure IoT systems by addressing critical challenges and fostering the adoption of decentralized approaches in key management.

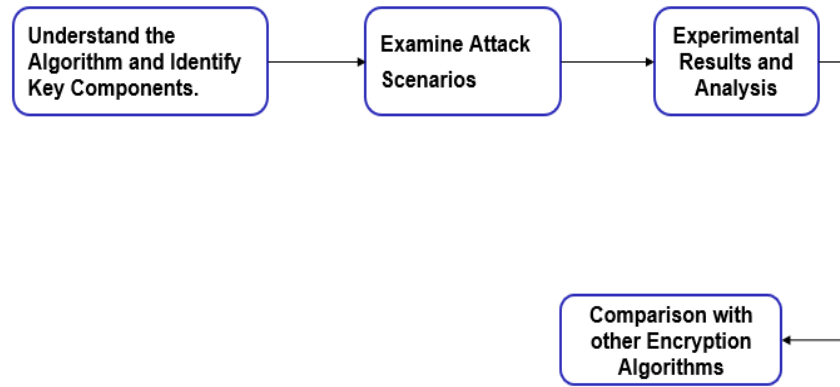


Fig. 3.1: Methodology of analyzing the DLGKM-AC algorithm

From Fig. 3.1, it can be observed that on the process and the steps that is to be handled to analyze the Decentralized Key Distribution Algorithm and implementation of the same in IoT devices.

3.1. UNDERSTANDING THE ALGORITHM & IDENTIFICATION OF THE KEY COMPONENTS

To understand a cryptographic algorithm effectively, break it down into key components through a detailed analysis. This involves dissecting the algorithm into fundamental building blocks and gaining insights into its operational logic, strengths, and weaknesses. This focused examination allows for a nuanced comprehension of the algorithm's functionality and security, forming the basis for mastery in cryptographic understanding.

3.2. EXAMINE ATTACK SCENARIOS

To effectively engage with a cryptographic algorithm, it is imperative to undertake a comprehensive understanding of its intricate details and systematically identify the key components that constitute its structure. This process involves a meticulous examination and deconstruction of the algorithm, dissecting it into its fundamental building blocks and constituent elements. By scrutinizing the algorithm at this granular level, one can gain insights into the underlying mechanisms, functions, and relationships of each key component, facilitating a nuanced comprehension of how the cryptographic process unfolds. In essence, the journey towards mastering a cryptographic algorithm commences with a diligent analysis that unveils its inner workings and discerns the pivotal elements that collectively contribute to its cryptographic functionality.

3.3. EXPERIMENTAL RESULTS AND ANALYSIS

In this study, we present experimental findings regarding the performance of the adapted DLGKM-AC scheme using MATLAB. We evaluate the efficiency of DLGKM-AC concerning storage, computation, and communication overheads associated with rekeying processes. Specifically, we focus on the additional

signaling load required for rekeying transmission following join/leave events. Furthermore, we conduct a comparative assessment between the proposed DLGKM-AC scheme and two alternative key management solutions designed for facilitating access control between subscribers and publishers. Our analysis aims to provide insights into the strengths and weaknesses of DLGKM-AC in comparison to existing approaches, shedding light on its suitability and effectiveness in dynamic IoT environments.

3.4. COMPARISON WITH OTHER ENCRYPTION ALGORITHMS

Compare the encryption algorithm with others to assess its strengths and weaknesses. This comparative evaluation aids in making informed decisions about selecting the most appropriate encryption method for a given scenario.

CHAPTER 4 – PROPOSED MODEL

The DLGKM-AC system is designed to efficiently distribute content to eligible subscribers while addressing concerns about single points of failure [8]. It introduces a hierarchical structure consisting of a central Key Distribution Center (KDC) and multiple Sub Key Distribution Centers (SKDCs) to oversee subscriber groups and manage membership changes effectively. Key management responsibilities are decentralized across SKDCs, enhancing system efficiency in terms of computation and communication. The KDC oversees device groups, while each SKDC handles user groups, improving scalability for DLGKM-AC.

In contrast, the Logical Key Hierarchy (LKH) scheme employs a tree structure for key distribution, reducing communication costs by multicasting multiple key-encryption keys logarithmically for each group of devices [9]. In this structure, devices are situated at the leaf nodes of the tree, with a central control center, the KDC, as mentioned in Fig 4.1. By managing the keys in the virtual tree. Each leaf node shares a secret key with the KDC, while the root holds the Group Key (GK), and internal nodes hold Key Encryption Keys (KEKs). Devices within the same subtree rooted to a specific internal node share the same KEKs,

collectively forming a Path Key (PKs) for efficient group key updates. In a complete tree with n devices, each device stores a logarithmic number of keys based on the total number of devices.

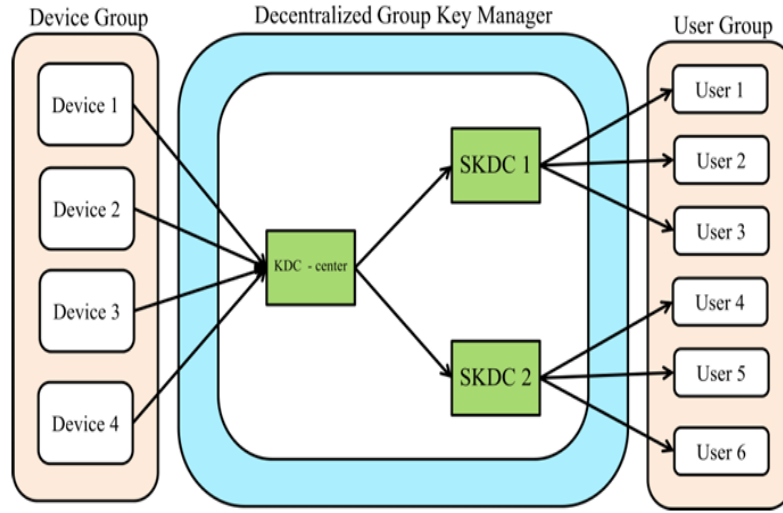


Fig. 4.1: Proposed System model

4.1. KEY GENERATION ALGORITHM

The Chinese remainder theorem (CRT) is a mathematical principle utilized to resolve simultaneous linear congruences with coprime moduli. In the realm of group key management (GKM) based on CRT, the central idea involves generating a single master key alongside multiple slave keys. This configuration allows messages encrypted with the master key to be decrypted by all authorized slave keys. The key generation process, which relies on the Generalized Chinese Remainder Theorem (GCRT), is facilitated by the Master Key Generation Algorithm. Both the Key Distribution

Center (KDC) and the Sub Key Distribution Centers (SKDCs) employ this algorithm to generate a master key for themselves and multiple slave keys for the SKDCs and user groups respectively under their jurisdiction. The Key Generation algorithm proceeds through a series of steps outlined in Fig. 4.2.

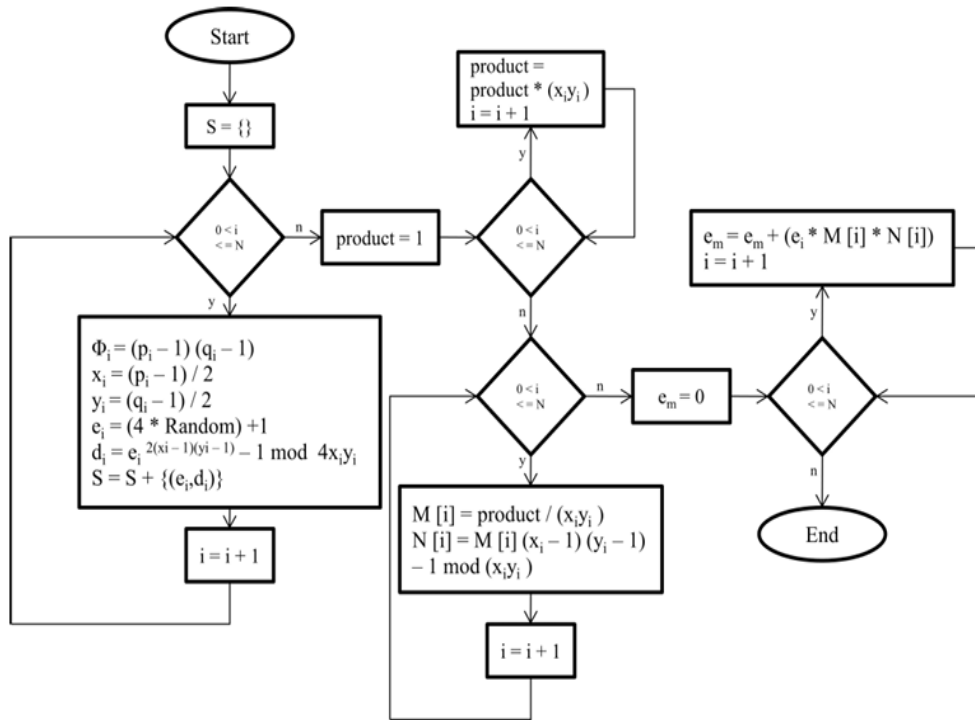


Fig. 4.2: Flow Chart of Key Generation Algorithm

When a new SKDC or user joins the system, the KDC or SKDC executes the Key Generation Algorithm to generate a new slave key for the newcomer and updates its own master key accordingly. The KDC establishes secure channels with devices and users, creating Device Groups (DGs) and assigning User Groups

(UGs) to SKDCs. An empty set is initialized to collect the slave key pairs, with a single master key (e_m) and multiple slave key pairs ($S\{\}$) being produced. The slave key generation process is iterated N times, as depicted in Fig.4.2. Once the slave keys are generated, the product is initialized to 1 and e_m to 0. The master key e_m is derived by computing the product of $M[i]$ and $N[i]$ for N iterations, where N represents the maximum number of slave keys provided by the SKDC, e_m signifies the master key, and $\{(e_i, d_i) ; 1 \leq i \leq N\}$ represents the set of slave keys (N public-private key pairs).

4.2. MASTER KEY ENCRYPTION

In the system architecture, users are granted access to multiple Data Groups (DGs) for data retrieval. To streamline this process, each user is furnished with all Traffic Encryption Keys (TEKs) corresponding to the DGs they are subscribed to. This approach serves to mitigate redundant subscriptions to the same DGs, which could otherwise escalate overhead during user subscription changes. Hence, effective management of group communication with users is paramount to minimize the overhead incurred when updating TEKs following each join or leave event.

In response to this challenge, we introduce the concept of Master Key Encryption (MKE), a key management scheme rooted

in Group Controlled Re-Keying Trees (GCRT). MKE facilitates the decryption of a message encrypted by an encryption key using multiple decryption keys. The fundamental principle of the MKE scheme revolves around generating a master key and several slave keys. By leveraging MKE, the rekeying costs associated with symmetric cryptography can be mitigated, thus enhancing the efficiency of key management within the system.

4.3. SCHEME DESCRIPTION

DLGKM-AC comprises three essential layers. The upper and lower layers represent clusters of devices (DGs) and users (UGs), respectively. In contrast, the middle layer denotes the decentralized controller, referred to the Key Distribution Center (KDC), responsible for managing keys both inter-group and intra-group.

i) Device Groups (DGs): Within the DLGKM-AC framework tailored for IoT environments, a predetermined set of Device Groups (DGs) is established based on criteria such as functionality, security levels, and geographical localization. Upon the integration of a new IoT device into the system, it is assigned to a specific existing DG. Communication within each DG is facilitated using the Logical Key Hierarchy (LKH).

ii) User Groups (UGs): DLGKM-AC for IoT environments organizes User Groups (UGs) based on user preferences and reservation periods. Each user is allocated to one of these UGs, with encryption keys distributed within each UG utilizing the Master Key Encryption (MKE) technique.

iii) Decentralized Group Key Manager: DLGKM-AC for IoT environments adopts a decentralized architecture comprising a central Key Distribution Center (KDC) and several Sub Key Distribution Centers (SKDCs). The number of SKDCs is determined by factors such as storage capacity, computational capabilities, and user population. The KDC acts as the primary server, connecting publishers to the system and overseeing key updates within DGs. Additionally, a backup server within the KDC maintains the latest key versions, periodically transmitting updates to the backup post-rekeying. Furthermore, SKDCs manage group communication within UGs, where users frequently join and leave the system. This decentralized approach, utilizing SKDCs, helps distribute the workload from the KDC. Each SKDC manages multiple user groups based on user localization, mitigating concerns regarding single-point failure (SKDC failure) and ensuring system scalability.

4.4. IDENTIFICATION OF ATTACKS ON DLGKM

In an era where digital security is paramount, the identification and mitigation of vulnerabilities in cryptographic algorithms are critical tasks. One such vulnerability lies in attacks susceptible to the DLGKM-AC algorithm. As cyber threats evolve, understanding and addressing weaknesses in cryptographic systems become increasingly urgent. The DLGKM-AC vulnerability underscores the ongoing arms race between security measures and malicious actors, highlighting the necessity for constant vigilance and innovative solutions in safeguarding sensitive data and digital infrastructures. The DLGKM-AC is vulnerable to the below-mentioned attacks.

*i) **Replay attack:*** In the realm of cybersecurity, replay attacks represent a formidable threat to the integrity and security of communication systems. This insidious form of cyber exploitation involves an attacker intercepting data packets exchanged between legitimate parties during a communication session. Once captured, the attacker stores these packets for later use. Subsequently, the attacker resends the intercepted data to one or more parties involved in the communication, exploiting the trust established between them. Since the packets contain legitimate information,

unsuspecting recipients may accept them as genuine and process them accordingly. The repercussions of a successful replay attack can be severe, ranging from unauthorized access to sensitive information to fraudulent transactions and manipulation of system behavior. Detecting and preventing replay attacks demands robust security measures, including techniques such as message authentication codes (MACs), digital signatures, timestamps, and sequence numbers to verify the integrity and freshness of data. Moreover, implementing secure communication protocols, employing encryption to safeguard data confidentiality, and maintaining vigilant monitoring of network traffic are essential countermeasures against this persistent threat. Understanding the intricacies of replay attacks and fortifying systems against them are crucial steps in upholding the resilience and reliability of modern communication infrastructures.

ii) Denial of Service (DoS): A Denial of Service (DoS) attack represents a perilous and pervasive threat, aiming to disrupt the normal functioning of a targeted system, network, or service. Operating on the premise of overwhelming the target with an excessive volume of requests, a DoS attack exhausts the resources, bandwidth, or processing capacity, rendering the system incapable

of responding to legitimate user requests. This nefarious strategy aims to impede access to critical resources, leading to service downtime and adversely impacting businesses, organizations, or individuals dependent on the targeted infrastructure. As the attack overwhelms and saturates the target's capacity to handle incoming requests, genuine users are denied access to essential services, resulting in a loss of productivity, revenue, and trust. DoS attacks can manifest in various forms, including network-based floods, application layer attacks, and more sophisticated Distributed Denial of Service (DDoS) assaults orchestrated by a network of compromised computers. Countering these attacks necessitates a multifaceted approach, encompassing robust network security measures, traffic filtering, and, in the case of DDoS attacks, coordinated efforts to identify and neutralize malicious traffic from multiple sources. The continual evolution of DoS attack techniques underscores the importance of ongoing research and innovation in developing resilient defenses to safeguard against this ever-present threat.

One drawback of DLGKM-AC is its susceptibility to replay attacks. In such attacks, an intruder intercepts a genuine message and later replays it to gain unauthorized access or an unfair

advantage. In the context of DLGKM-AC, this vulnerability arises when an attacker intercepts a message containing a group key and replays it subsequently to enter a secure group communication session. Another limitation of DLGKM-AC is its vulnerability to denial-of-service (DoS) attacks. In a DoS attack, assailants endeavor to render a computer or network unavailable to legitimate users. In the case of DLGKM-AC, attackers may flood a device with numerous invalid group keys, causing it to become overwhelmed and inaccessible to genuine users.

To mitigate replay attacks, DLGKM-AC could be modified to use sequence numbers in group key messages. This would allow devices to detect and reject replayed messages. To mitigate DoS attacks, DLGKM-AC could be modified to use a rate-limiting mechanism to limit the number of group key messages that a device can receive in each period. In addition to these specific modifications, DLGKM-AC could also be modified to make it more efficient and to support a wider range of IoT devices. For example, DLGKM-AC could be modified to use a different tree structure or to support different types of symmetric encryption and digital signatures. So, by adding sequence numbers to the group keys in the DLGKM-AC algorithm, can prevent replay attacks which can be

implemented using MATLAB which will be simulated by us and shown in the future. Also, the performance of the modified algorithm that is immune to replay attacks will be analyzed, as ways to reduce the computational and communication overhead.

CHAPTER 5 – SIMULATION RESULTS

5.1. INTRODUCTION

The registration of users in user groups and the Key generation code are successfully simulated using Python. The results are discussed in this section.

5.2. INITIAL INTERFACE

For the initial registration of the users into the user groups and logging in into the specific window, a Python interface is created. It includes the login and register buttons, which is clearly shown in Fig 5.1. Tkinter module is used to create such interfaces. When the Register button is triggered, the user is prompted to register, using his name, the required device group and a password. The details get uploaded on an excel sheet.

When the Login button is triggered, it opens a login window in which many functions are performed, such as encryption and decryption. It obtains details like name, device group and password, verifies it with the database present in the excel sheet and logs in.

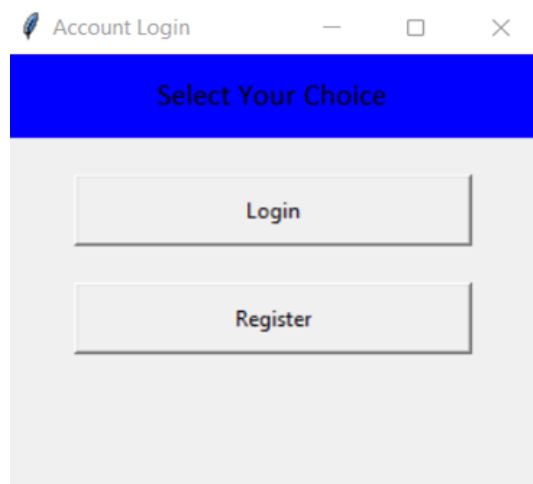


Fig 5.1: GUI to Register users and Login

5.3. REGISTRATION OF USERS

For the initial registration of the users, the register button is used. If the user is already present in the system, then registration cannot be done, and login button should be used. If the user is new to the system, he is prompted to register first, as shown in Fig 5.2. User public key and User private key are generated using Key generation algorithm. The corresponding Device Group public key and private key is also generated using the Key Generation Algorithm. The registered details of the user, and the corresponding keys are stored in an excel sheet as shown in Table 5.1.

When a new user joins, he registers and the corresponding Device Group public keys and private key is again generated and updated to all the users under the specific device group, as shown in Table 5.2.

Fig 5.2: GUI to Register user

	A	B	C	D	E	F	G	H
	Name	Device Group	Password	User Public Key	User Private Key	Device Group Public Key	Device Group Private Key	
1								
2	pranav	1	pv	9	1520	17	720	
3	pranav new	1	pv	17	720	17	720	
4	gokul	2	gokul	29	736	29	736	
5								
6								
7								

Table 5.1: Excel sheet with new user's details

	A	B	C	D	E	F	G	H
	Name	Device Group	Password	User Public Key	User Private Key	Device Group Public Key	Device Group Private Key	
1								
2	pranav	1	pv	9	1520	17	720	
3	pranav new	1	pv	17	720	17	720	
4	gokul	2	gokul	29	736	41	1680	
5	gokul new	2	gokul	11	1360	41	1680	
6								

Table 5.2: Excel sheet with new Registered user's details
and updated Device group keys – User joining

5.4. LOGIN PAGE FOR USERS

The login button opens a new interface, as shown in Fig 5.3., which helps the user to login into the main window. If any one of

the details asked is entered wrongly, then an error pops up, asking them to check the details. If the name is not present, it gives a pop up saying that the user is not found, as shown in Fig 5.4. If the Device group is entered wrongly, it pops up saying, Invalid Device Group, as shown in Fig 5.5. If the password is entered incorrectly, it pops up saying Invalid Password, as shown in Fig 5.6. Once all the details are correct, and the username is already registered, it successfully logs in, into the main window with a Login Successful pop up, as shown in Fig 5.7.

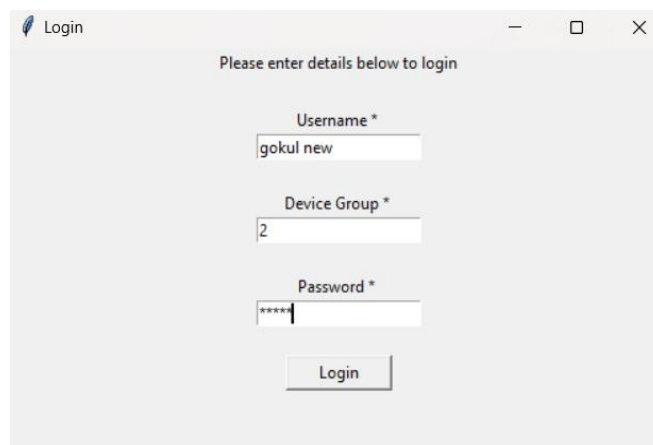


Fig 5.3: GUI to Login user

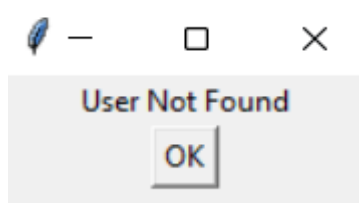


Fig 5.4: User not found Pop-up

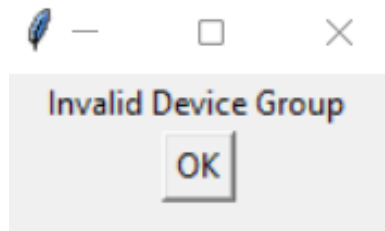


Fig 5.5: Invalid Device group Pop-up

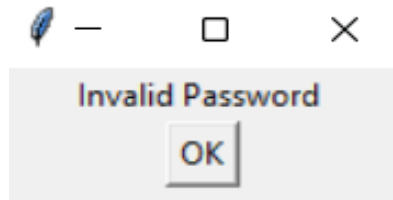


Fig 5.6: Invalid Password Pop-up

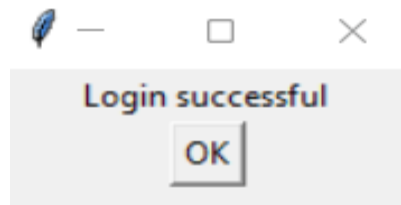


Fig 5.7: Login successful Pop-up

The main window opens, as shown in Fig 5.8. It displays the Name and the Device Group of the user who logs in. It has 6 buttons, namely the Upload button, Encrypt Button, Decrypt Button, Download Button, Delete User Button and Exit Button. It has a blank text space, which displays the text file being uploaded, encrypted and decrypted. Exit closes the login main window.

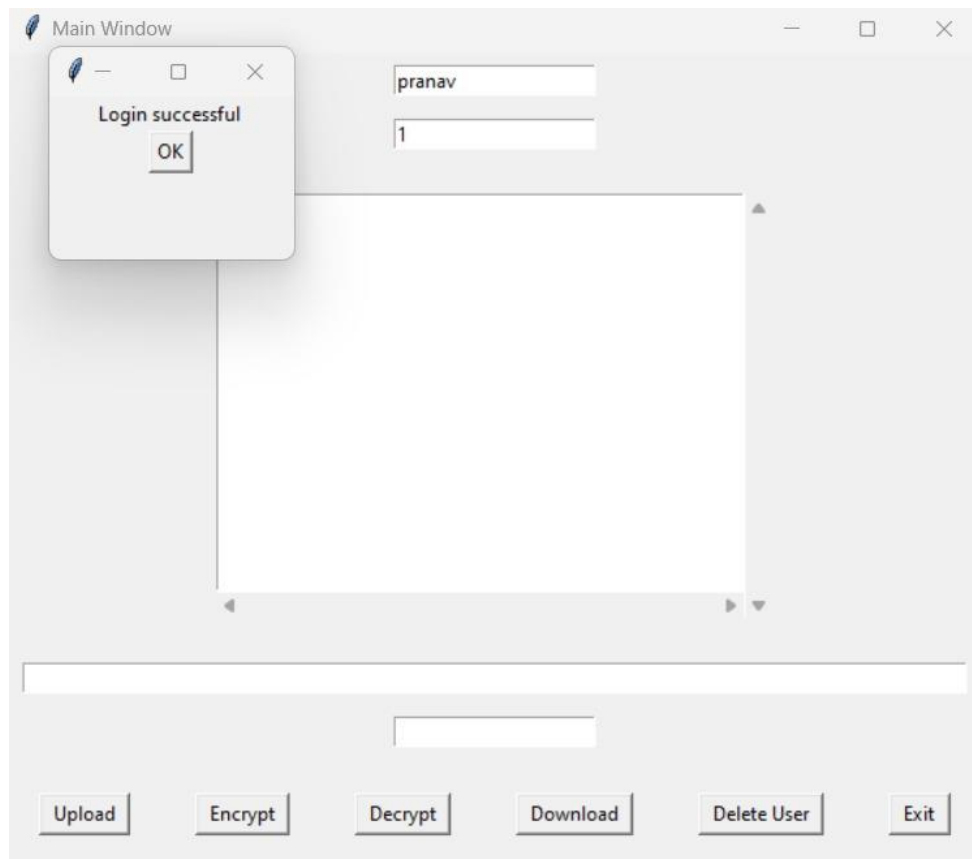


Fig 5.8: GUI of the Main Window

5.5. ENCRYPTION OF TEXT FILES

The uploaded file is encrypted when the encrypt button is triggered in the main window. The encrypted text is displayed in the blank text space, as shown in Fig 5.9. Encryption is done using an inbuilt RSA function in python, using public key and private key.

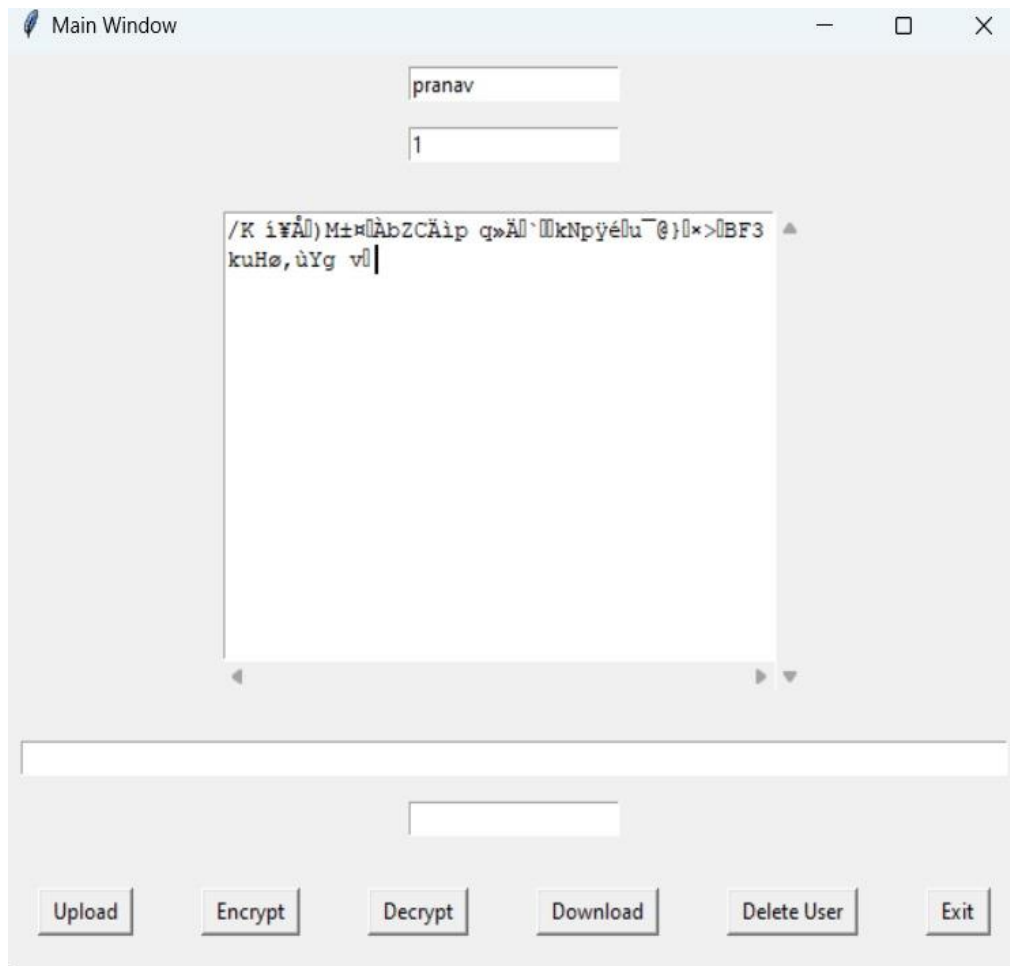


Fig 5.9: GUI window after encrypting the required text file

5.6. DECRYPTION OF TEXT FILES

When the decrypt button is triggered in the main window, A pop up appears, asking the user's private key, as shown in Fig 5.10. If the private key entered matches with the database, then another pop up appears, asking the device group's public key, as shown in Fig 5.11. If the public key entered matches with the database, then decryption occurs. The decrypted text is displayed as in Fig 5.12.

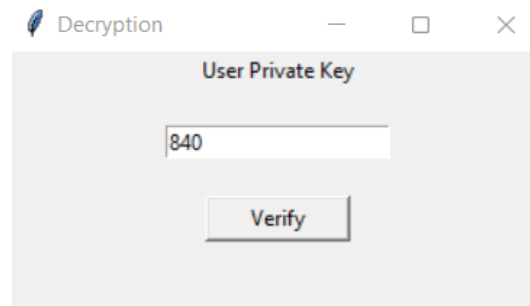


Fig 5.10: Pop up asking user's private key

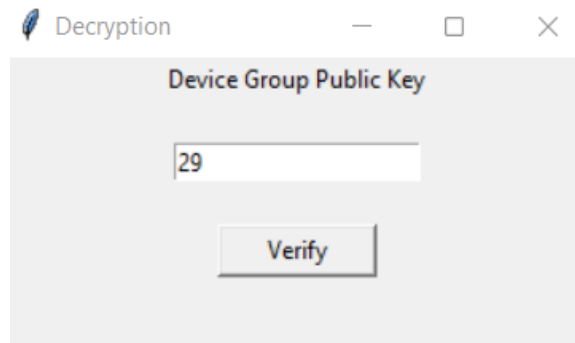


Fig 5.11: Pop up asking Device group's public key

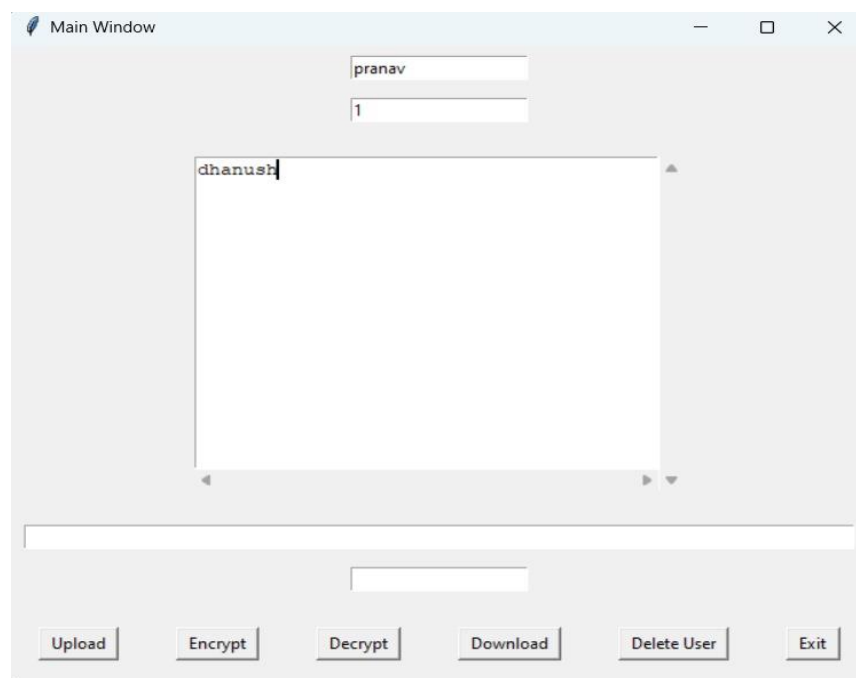


Fig 5.12: GUI window after decrypting Text file

5.7. DOWNLOADING DECRYPTED FILES

When the download button is triggered in the main window, A Browsing window appears, asking for the text file name to be downloaded, as shown in Fig 5.13. The file is downloaded and the file path is updated in the main window, as shown in Fig 5.14. The downloaded file is shown in Fig 5.15.

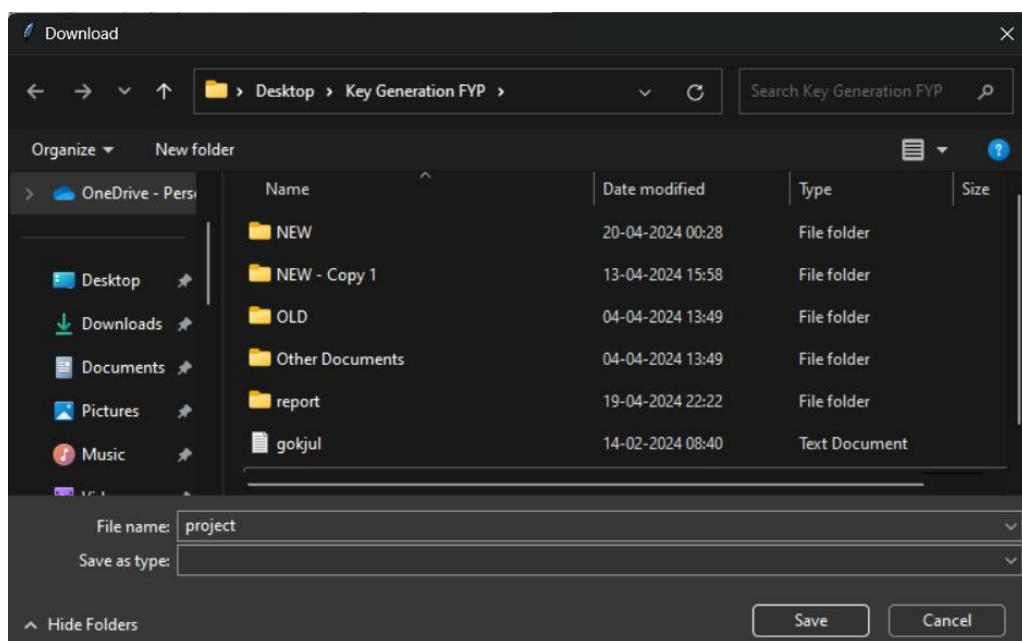


Fig 5.13: Browsing window after triggering the Download button



Fig 5.14: GUI window after downloading the decrypted text file

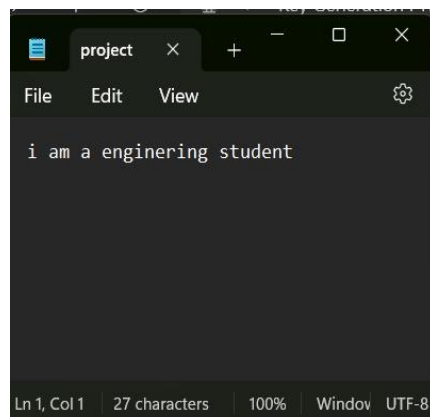


Fig 5.15: Plain Text file downloaded

5.8. ANALYSIS OF KEYGEN ALGORITHM

Here, to check the working of the algorithm, we have considered $N = 1$, have taken random prime integers (p,q) between 25 to 50 and run the algorithm. As a Result, we have displayed 2 slave key pairs, 1 master key, the total time taken to run the algorithm, as shown in Fig 5.16.

```
S:
[['0b101', '0b1001110000'], ['0b101', '0b1100011000'], ['0b1001', '0b1010000']]

em:
0b100101101110110100110101011001100101000010000
|
Execution time:
0.014346799987833947

| | 3 function calls in 0.000 seconds

Ordered by: standard name

ncalls  tottime  percall  cumtime  percall filename:lineno(function)
| 1 0.000 0.000 0.000 0.000 <string>:1(<module>)
| 1 0.000 0.000 0.000 0.000 {built-in method builtins.exec}
| 1 0.000 0.000 0.000 0.000 {method 'disable' of '_lsprof.Profiler' objects}

5
624
```

Fig 5.16: Analysis of KeyGen Algorithm

The Slave keys and Master key is printed in binary digits. The execution time is printed using an inbuilt python function. The CProfile of the algorithm is also printed.

5.9. SEQUENCING TO AVOID REPLAY ATTACKS

Sequence numbering is employed as a countermeasure against replay attacks, ensuring data integrity and security in communication protocols. By assigning unique sequential identifiers to each transmitted data packet or message, sequence numbering enables the recipient to verify the freshness of received data. This prevents adversaries from replaying previously captured messages to gain unauthorized access or disrupt the system.

```
1 class MessageSender:
2     def __init__(self):
3         self.sequence_number = 0
4
5     def generate_sequence_number(self):
6         self.sequence_number += 1
7         return self.sequence_number
8
9     def send_message(self, message):
10        sequence_number = self.generate_sequence_number()
11        message_with_sequence = f"{sequence_number}:{message}"
12        print("Sending message:", message_with_sequence)
13
14 class MessageReceiver:
15     def __init__(self):
16         self.received_sequence_numbers = set()
17
18     def receive_message(self, message):
19        sequence_number, payload = message.split(':', 1)
20        sequence_number = int(sequence_number)
21        if sequence_number in self.received_sequence_numbers:
22            print("Replay attack detected! Message ignored.")
23            return
24        print("Received message:", payload)
25        self.received_sequence_numbers.add(sequence_number)
```

Fig 5.17: Sequence Numbering

The block of code used to integrate sequencing in the proposed model is represented in Fig 5.17. Essentially, sequence numbering adds a temporal dimension to data integrity, allowing the recipient to

discern between old and new messages and reject any duplicates. This technique is fundamental in safeguarding against replay attacks, enhancing the resilience of communication systems without relying solely on encryption or authentication mechanisms.

CHAPTER 6 – ANALYSIS AND EVALUATION

6.1. SECURITY ANALYSIS

Security is required to protect the session keys against being compromised even when the server's private key may be vulnerable. This section deals with the security analysis of the proposed scheme in terms of Forward Security and Backward Security [10].

6.1.1. FORWARD SECURITY

Forward Security refers to the guarding against future compromises of the past sessions. In other words, when a node (user) leaves the network, it must not read any future messages after its departure. Here the key management scheme between SKDC and users provides forward security. The user who left the group, will not be able to access the ongoing communication. Consider user U_j leaves the group. Hence the key pair (e_j, d_j) should be revoked when the user leaves. Thus, the new formed master key of the UG_k satisfies Eqn. (6.1.1) & Eqn. (6.1.2),

$$P_m^{e'} \equiv P_i^e \text{ mod}(p_i q_i) \quad (6.1.1)$$

$$C_m^{d'} \equiv C_i^d \text{ mod}(p_i q_i); \forall i \in [1, r_K], i \neq j \quad (6.1.2)$$

The plaintext P is encrypted with the new master key to generate the ciphertext C^* at the data source. Upon receiving the new ciphertext C^* , each user within the group can decrypt it using their private key. Despite having knowledge of the old keys, a departing user cannot decrypt the original plaintext from the ciphertext. Therefore, forward security is preserved between the SKDC and the user group when a user exits the group. The key management mechanism between the KDC and IoT devices ensures forward security, preventing a device that has left the group from accessing ongoing communications. Suppose A_1 represents an adversary collaborating with an IoT Device D_j within the device group DG_k . In this scenario, A_1 may obtain all information stored in the IoT devices. However, this scheme updates the new group keys within the device group, preventing A_1 from decrypting the message and accessing the group key.

6.1.2. BACKWARD SECURITY

Backward Security refers to guarding against past compromises of future sessions. In other words, when a new node (user) joins the network, it must not read any previously transmitted messages after its arrival. Here the key management scheme between SKDC and users provides backward security. The user who

joins the group will not be able to access the previous communications. Consider user U_j joining the group with the key pair (e_j, d_j) . Thus, the newly formed master key of the UG_k satisfies Eqn. (6.1.3) & Eqn. (6.1.4),

$$P_m^e \equiv P_i^e \text{ mod}(p_i q_i) \quad (6.1.3)$$

$$C_m^d \equiv C_i^d \text{ mod}(p_i q_i); \forall i \in [1, r_K], i \neq j \quad (6.1.4)$$

The plaintext P undergoes encryption with the new master key, resulting in the ciphertext C at the data source. Upon receiving this new ciphertext C , each user within the group can decrypt it using their private key. Even though a new user possesses knowledge of the current keys, they are unable to decrypt the previous plaintext from the ciphertext. Consequently, backward security is upheld between the SKDC and the user group once the user becomes a part of the group. Similarly, the key management system between the KDC and IoT devices ensures backward security. When a device exits the group, it loses access to past communications, as new group keys are generated. Consequently, the old keys cannot be inferred by the new device D_j , thereby preventing the newly joined device from gaining insights into previous group keys and communications.

6.2. PERFORMANCE ANALYSIS

This section deals with the performance analysis of the proposed scheme in terms of overhead. Here we assume that the IoT devices are equally distributed in each device group.

6.2.1. STORAGE OVERHEAD

The Storage Overhead can be defined as the memory capacity needed to store keys within the system. In our architecture, we formulate the storage overhead for both individual users within a user group and for devices within a device group. Each user within a user group is equipped with multiple symmetric keys (SK) and an asymmetric key (AK). We quantify the storage requirements for keys per user using Equation (6.2.1).

$$SO_{U \in UG_x} = AK + (\sum_{i=1}^M A_{i,b} + 1) SK \quad (6.2.1)$$

6.2.2. COMPUTATION OVERHEAD

Computational overhead refers to the collective time spent on encryption and decryption, along with the processing demands. These tasks occur across servers, users, and devices following each member's joining or leaving actions within the system.

6.2.2.1. WHEN A USER JOINS A USER GROUP

The SKDC initiates updates to its master key and produces a slave key for newly joining users in the group. Upon joining, a new user requires only one symmetric decryption to acquire both the slave key and the device group key relevant to the subscribed device group. Existing users undergo a hashing process to update their keys, while devices undergo two hashing operations to derive their new device group keys.

6.2.2.2. WHEN A USER LEAVES A USER GROUP

Following a user's departure from the group, the SKDC updates its master key. To disseminate this update, the remaining user groups undergo one asymmetric decryption and one symmetric decryption to obtain the updated information. Similarly, the devices subscribed to these user groups perform one symmetric decryption to access the details regarding the device group keys.

6.2.2.3. WHEN A DEVICE JOINS A DEVICE GROUP

Old devices within the departed device group need to execute one hash function to update their device group key. Conversely, new devices simply decrypt the message dispatched by the KDC to acquire the new device key.

6.2.2.4. WHEN DEVICE LEAVES A DEVICE GROUP

The remaining devices undergo one symmetric decryption to obtain the new group key. Users subscribed to the departing groups, however, are not required to perform any additional computations.

6.2.3. COMMUNICATION OVERHEAD

As the IoT devices often operate under resource-constrained environments, such as limited processing power and energy reserves, the overhead introduced by cryptographic operations and communication exchanges can severely impact the overall performance and efficiency of the system. Therefore, mitigating communication overhead while maintaining robust security measures remains a crucial objective in the design and implementation of decentralized lightweight group key management solutions for IoT devices. To assess the communication burden of the newly introduced DLGKM-AC in the context of IoT, we examine the quantity of update key messages transmitted upon a user's arrival and departure.

Events	Communication Overhead
User leaves	<ol style="list-style-type: none"> 1. SKDC broadcasts the new keys to the subgroups. 2. KDC sends messages to the devices.
User joins	<ol style="list-style-type: none"> 1. SKDC unicasts a message to the new user. 2. All users update their keys.
Device joins	<ol style="list-style-type: none"> 1. KDC unicasts a message to the new device. 2. KDC broadcasts new keys to the subscribers.
Device leaves	<ol style="list-style-type: none"> 1. KDC indicates the users to update their keys as the device leaves.

Table 6.1: Communication Overhead

6.2.4. PERFORMANCE EVALUATION

The work contrasts the newly proposed DLGKM-AC scheme with an alternative key management solution tailored for controlling access between subscribers and publishers. To support individual device management, the traditional LKH method constructs a separate LKH tree for each user group. Unlike our proposed approach, where only one tree's KEK needs handling, users in the traditional scheme must store both encryption group keys and multiple KEKs, increasing their storage burden. Conversely,

devices in this method require minimal storage as they aren't grouped. When a new user joins and subscribes to devices, all existing trees must adjust, necessitating updates to group keys and distributing new KEKs to affected users. If an existing user subscribes to additional devices, this overhead multiplies with each device. In contrast, our method requires users to store only one set of KEKs per subgroup tree.

In update scenarios, new users only need to update KEKs in one tree. Existing users need updates only if their subscription matches the new user's, reducing update complexity. Although asymmetric decryption overhead is higher than symmetric computation, as users leave, the need to update device group keys outweighs this overhead, especially with a growing number of devices.

A decentralized architecture helps alleviate the computational burden that arises after a user leaves a network. In contrast, utilizing Subgroup Key Distribution Centers (SKDCs) can mitigate the computational overhead involved in updating keys within an Internet of Things (IoT) environment. Unlike [11] scheme, where the computational overhead increases with the number of users in each User Group (UG), our proposed algorithm for managing communication within UGs ensures low computational costs even

with a high number of users. We demonstrate the efficiency of our group key updating scheme based on the Chinese Remainder Theorem (CRT), showing that our solution requires less time for key updating compared to traditional schemes like LKH, particularly when a user is revoked. Furthermore, our scheme is less impacted by the number of Distribution Groups (DGs) users are subscribed to compared to previous approaches, thanks to the use of subgroup controllers (SKDCs) which streamline key updates and reduce computational burdens for end-users.

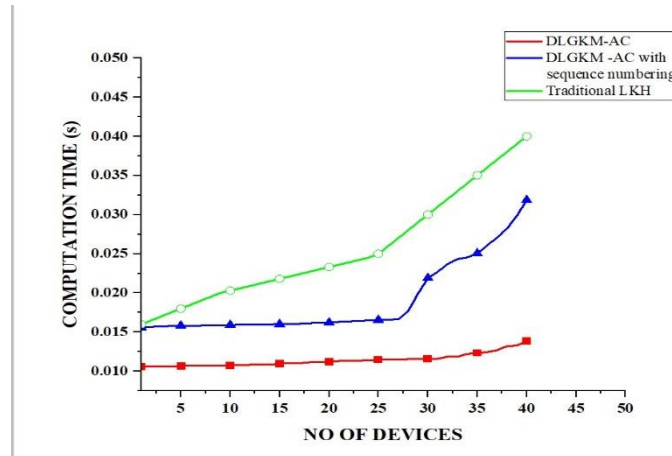


Fig 6.1: Computation Overhead

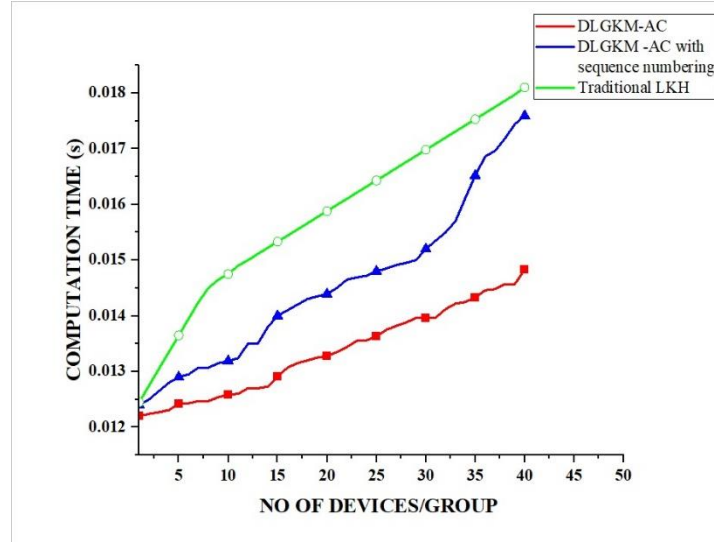


Fig 6.2: Computation Overhead per Group

Fig 6.2. presents the update overhead on a new user who joins the same user group. In contrast to [11], our scheme incurs minimal computational overhead. Specifically, a new user only needs to decrypt received messages to access information, whereas in [11], new users must compute the device keys to which they are subscribed.

CHAPTER 7 – CONCLUSION

Implementing a decentralized key distribution algorithm for IoT devices is a crucial stride toward guaranteeing the security, privacy, and efficiency of interconnected devices in today's digitally interconnected world. The challenges presented by the extensive scale of IoT deployments, and the vulnerabilities linked with centralized key management systems underline the need for a decentralized approach. This approach enables devices to securely exchange keys without depending on a single point of control. Thus, the DLGKM-AC algorithm is being used to encrypt data using different group keys and the approach to modify the above algorithm to avoid replay attacks is defined. The decentralized lightweight group key management scheme proposed for IoT devices represents a significant advancement in addressing the dynamic access control challenges prevalent in IoT environments. By decentralizing key management tasks and leveraging lightweight cryptographic techniques, the proposed scheme ensures robust security without imposing undue overhead on resource constrained IoT devices. Furthermore, the scheme's adaptability to evolving IoT environments underscores its relevance and potential for widespread adoption in various IoT applications. Overall, this

research contributes significantly to enhancing the security and resilience of IoT ecosystems, paving the way for safer and more reliable deployment of IoT devices in real-world scenarios. Further, future works to be done are to investigate adaptive sequencing mechanisms or incorporate machine learning approaches to dynamically adjust the modified DLGKMAC algorithm, enhancing its resistance against evolving replay attack strategies. To conduct a thorough performance evaluation of the modified DLGKM-AC algorithm using MATLAB. Assessment of computational efficiency, storage requirements, and overall performance compared to other prominent algorithms by considering factors such as processing speed, memory utilization, and energy consumption. Also, exploration of real-world deployment scenarios to assess the modified DLGKM-AC algorithm's performance under diverse conditions by conducting simulation studies and possibly hardware implementations to validate its efficacy in various IoT environments.

By addressing these areas, future work aims to not only refine the algorithm's resilience against replay attacks but also provide a comprehensive understanding of its performance characteristics in comparison to existing algorithms.

REFERENCES

- [1] Abdmeziem.M.R. ,Tandjaoui.D, and I. Romdhani, "A decentralized batch-based group key management protocol for mobile Internet of Things (DBGK)," in *Proc. IEEE Int. Conf. Comput. Inf. Technol. Ubiquitous Comput. Commun. Depend. Auton. Secure Comput. Pervasive Intell. Comput.*, Liverpool, U.K., 2015, pp. 1109–1117.
- [2] M. Egorov and M. Wilkison, "NuCypher KMS: Decentralized key Management system," NICS Lab, Universidad de Malaga, Spain, 2017.
- [3] Dammak. M, Senouci. S.M, Messous. M. A, Elhdhili. M. H, & Gransart. C, “Decentralized Lightweight Group Key Management for Dynamic Access Control in IoT Environments”, IEEE Transactions on Network and Service Management, vol. 17, September 2020.
- [4] Mamvong. J.N, Goteng.G.L, Zhou. B,Gao. Y, “Efficient Security Algorithm for Power-Constrained IoT Devices”, IEEE Internet of Things Journal, vol. 8, April 2021.
- [5] Bonde. S. Y, Bhadade. U. S. “ Analysis of Encryption Algorithms (RSA, SRNN and 2 key pair) for Information Security”, IEEE Encryption Algorithm, 2017.
- [6] Lavanya. R, Sundarakantham K, Shalinie. S. M, “Cost Effective Decentralized Key Management Framework for IoT”, Tech Science Press CSSE, vol. 41,2022.
- [7] Zhang. W., Zhang. J, Zhang. X, “ A Lightweight Authenticated Group Key Management Scheme for the Internet of Things”,IEEE Cryptography Algorithm, 2018
- [8] Y. Kung and H. Hsiao, "GroupIt: Lightweight group key

management for dynamic IoT environments," *IEEE Internet Things J.*, vol. 5, no. 6, Dec. 2018

[9] A. S. Pande and R. C. Thool, "Survey on Logical Key Hierarchy for secure Group Communication," in *International Conference on Automatic Control and Dynamic Optimization Techniques (ICACGOT)*, 2016.

[10] M. Peyravian, S. M. Matyas, and N. Zunic, "Decentralized group key management for secure multicast".

[11] Y.Kung and H.Hsiao, "GroupIt: Lightweight group key management for dynamic IoT environments," IEEE Internet Things J., vol.5, no.6, pp.5155–5165, Dec.2018.

Final year project

ORIGINALITY REPORT

22%

SIMILARITY INDEX

17%

INTERNET SOURCES

19%

PUBLICATIONS

3%

STUDENT PAPERS

PRIMARY SOURCES

1

hal.archives-ouvertes.fr

Internet Source

11%

2

tel.archives-ouvertes.fr

Internet Source

2%

3

Submitted to SSN COLLEGE OF
ENGINEERING, Kalavakkam

Student Paper

2%

4

Maissa Dammak, Sidi Mohammed Senouci,
Mohamed Ayoub Messous, Mohamed
Houcine Elhdhili, Christophe Gransart.
"Decentralized Lightweight Group Key
Management for Dynamic Access Control in
IoT Environments", IEEE Transactions on
Network and Service Management, 2020

Publication

1%

5

Maissa Dammak, Sidi-Mohammed Senouci,
Mohamed Ayoub Messous, Mohamed
Houcine Elhdhili, Christophe Gransart.
"Decentralized Lightweight Group Key
Management for Dynamic Access Control in

1%