# ANALYTICAL MODELS FOR THE SCALABILITY OF DYNAMIC GROUP KEY AGREEMENT PROTOCOLS AND SECURE FILE SHARING SYSTEMS

by

Gökcan Çantalı

B.S., Computer Engineering, Boğaziçi University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2019

ANALYTICAL MODELS FOR THE SCALABILITY OF DYNAMIC GROUP KEY AGREEMENT PROTOCOLS AND SECURE FILE SHARING SYSTEMS

APPROVED BY:

| | |
|---|---|
| Prof. Cem Ersoy<br>(Thesis Supervisor) | ..... |
| Prof. M. Ufuk Çağlayan<br>(Thesis Co-supervisor) | ..... |
| Prof. Fatih Alagöz | ..... |
| Assist. Prof. Şerif Bahtiyar | ..... |

DATE OF APPROVAL: 18.01.2019

# ACKNOWLEDGEMENTS

To my beloved wife, Damlanur Keskin Çantalı: I cannot express how grateful I am for all your support and love.

Foremost, I would like to express my gratitude to Orhan Ermiş for his tremendous support and help to improve my work. You have never stopped giving me advices and you have always been patient while correcting my mistakes. I sincerely thank you for everything you have done for me.

I would like to express my deepest appreciation to my thesis co-advisor Prof. M. Ufuk Çağlayan, who has been an incredible mentor since my undergraduate studies and role model for me. Also, I would like to extend my deepest gratitude to my advisor Prof. Cem Ersoy for guiding me through my research and for providing me ideas whenever I needed them. My sincere thanks go to jury members: Prof. Fatih Alagöz and Assist. Prof. Şerif Bahtiyar for their participation in my thesis committee in their rare time and their constructive comments on my thesis.

My special thanks to my family, I am truly grateful to my parents, Nurten and Mustafa for everything they have done for me. I wish to thank them for supporting me through my whole life and for providing me their love, patience and encouragement.

I am grateful to all my colleagues in Boğaziçi University Computer Engineering Department. Especially, I owe special thanks to Gürkan Gür, Meriç Turan, Alper Kamil Bozkurt, Hasan Ferit Enişer and Can Güven for their support, discussions and encouragements during my MS studies. I also would like to thank Cihat Baktır, Barış Yamansavaşçılar, Samet Aytaç, Nurefşan Sertbaş, Serdar Metin, Alper Alimoğlu, Niaz Chalabianloo, Akif Emre, Hakan Selvi and Can Tunca for their friendship, support and understanding.

Last, but not the least, I would like to thank Levent Baş, İsmail Şahiner, Alper Taştemur, Feyzullah Alim Kalyoncu, Mert Değirmenci, Ümit Öztürk, Taha Metin Bayi, Eray Sezgin and Melih Mutlu for their friendship.

My deepest apologies go to those who I missed to mention here. I wish to thank them all for being always with me when I needed them.

# ABSTRACT


## ANALYTICAL MODELS FOR THE SCALABILITY OF DYNAMIC GROUP KEY AGREEMENT PROTOCOLS AND SECURE FILE SHARING SYSTEMS

Dynamic group key agreement protocols are cryptographic primitives to provide secure group communications in decentralized and dynamic networks. Such protocols provide additional operations to update the group key while adding new participants into the group and removing existing participants from the group without re-executing the protocol from the beginning. However, the lack of scalability emerges as one of the most significant issues of dynamic group key agreement protocols when the number of participants in the group increases. For instance, frequent participant join requests for large groups may cause an effect similar to a Distributed Denial of Service attack and violate the system availability due to the increase in group key update time. Therefore, analyzing the scalability of dynamic group key agreement protocols is crucial to detect conditions where the system becomes unavailable. In this thesis, we propose a formal performance model to evaluate the scalability of dynamic group key agreement protocols by using queueing models. We also extend our performance model for evaluating the scalability of secure file sharing systems that utilize group key agreement protocols. Moreover, we present a demonstrative use case to show the applicability of our performance model on an example group key agreement protocol and a secure file sharing system. Furthermore, we design attack models that can be used against communication systems and explain how the target system may be affected by a cyber attack.

# ÖZET

## DİNAMİK GRUP ANAHTAR ANLAŞMA PROTOKOLLERİNİN VE GÜVENLİ DOSYA PAYLAŞIM SİSTEMLERİNİN ÖLÇEKLENEBİLİRLİĞİ İÇİN ANALİTİK MODELLER

Dinamik grup anahtarı anlaşma protokolleri, dağıtık ve dinamik ağlarda güvenli grup iletişimi sağlayan temel kriptografik öğelerdir. Bu protokoller, sağladıkları fazladan operasyonlar ile gruba yeni katılımcılar eklendiğinde ya da mevcut katılımcılar gruptan ayrıldığında, protokolü en baştan çalıştırmadan grup anahtarının güncellenmesini sağlamaktadır. Ancak, grup katılımcılarının sayısı arttıkça, ölçeklenebilirlik eksikliği dinamik grup anahtarı anlaşma protokollerinin başlıca sorunlarından biri haline gelmektedir. Örnek olarak, büyük gruplara sık bir şekilde gelen katılımcı ekleme isteği Dağıtık Servis Dışı Bırakma saldırısına benzer bir etkiye sebep olarak, sistemin kullanılırlığının ihlal edilmesine yol açabilir. Bu sebeple, dinamik grup anahtarı anlaşma protokollerinin ölçeklenebilirlik analizinin yapılması, sistemin kullanılamaz noktaya geldiği durumların tespiti açısından çok önemlidir. Bu tezde, dinamik grup anahtarı anlaşma protokollerinin ölçeklenebilirlik değerlendirmesini kuyruk modelleri aracılığıyla yapan biçimsel bir başarım modeli öneriyoruz. Ayrıca, başarım modelimizi dinamik grup anahtarı anlaşma protokolleri kullanan güvenli dosya paylaşım sistemlerinin ölçeklenebilirlik analizini de kapsayacak şekilde genişletiyoruz. Buna ek olarak, modelimizin örnek bir dinamik grup anahtarı anlaşma protokolü ve bir güvenli dosya paylaşım sistemi üzerinde uygulanabilirliğini gösteren bir kullanım senaryosu sunuyoruz. Dahası, grup iletişim sistemlerine karşı kullanılabilecek saldırı modelleri tasarlayarak, hedef alınan sistemin bir siber saldırı altında nasıl etkilendiğini gösteriyoruz.

# TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . . iii
ABSTRACT . . . . . v
ÖZET . . . . . vi
LIST OF FIGURES . . . . . ix
LIST OF TABLES . . . . . xi
LIST OF SYMBOLS . . . . . xii
LIST OF ACRONYMS/ABBREVIATIONS . . . . . xiv
1. INTRODUCTION . . . . . 1
1.1. Motivation . . . . . 2
1.2. Contributions . . . . . 2
1.3. Thesis Outline . . . . . 3
2. MATHEMATICAL BACKGROUND AND LITERATURE OVERVIEW . . 5
2.1. Mathematical Background and Definitions for Queueing Theory . . . . 5
2.2. Group Key Agreement Protocols . . . . . 10
2.3. Secure File Sharing Systems . . . . . 11
2.4. A Comparison of Performance Evaluation Methods used in Secure Multi-Party Group Communication Systems . . . . . 14
3. ANALYTICAL MODELS FOR THE SCALABILITY ANALYSIS OF DYNAMIC GROUP KEY AGREEMENT PROTOCOLS . . . . . 17
3.1. Residence Model . . . . . 18
3.2. Join Model . . . . . 19
3.2.1. Single Join Operation . . . . . 20
3.2.2. Mass Join . . . . . 23
3.3. Leave Model . . . . . 27
3.4. Possible Scenarios for Analytical Models . . . . . 29
3.5. Attack Models . . . . . 31
4. ANALYSIS OF PROPOSED ANALYTICAL MODELS ON AN EXAMPLE GROUP KEY AGREEMENT PROTOCOL . . . . . 35
4.1. Analysis of the Single Join Operation . . . . . 36

4.2. Analysis of the Mass Join Operation . . . . . 38
4.3. Analysis of the join-PBC Operation . . . . . 40
4.4. Analysis of the Leave Operation . . . . . 42
4.5. Numerical Results for the Example KAP-PBC Use Case Scenario . . . 45
5. ANALYTICAL MODELS FOR THE SCALABILITY ANALYSIS OF SECURE FILE SHARING SYSTEMS WITH A DEMONSTRATIVE USE CASE SCENARIO . . . . . 49
5.1. Performance Model for Secure File Sharing System Operations by Using PFSS as a Demonstrative Use Case Scenario . . . . . 49
5.2. A Use-Case Scenario for Healthcare System . . . . . 51
5.3. Numerical Results for the PFSS Use-Case Scenario . . . . . 55
6. CONCLUSION . . . . . 59
6.1. Future Work . . . . . 60
REFERENCES . . . . . 61

# LIST OF FIGURES

Figure 3.1. An Overall View of the Proposed Performance Model for Dynamic Group Key Agreement Protocols . . . . . 17

Figure 3.2. The model for the single join operation . . . . . 21

Figure 3.3. Markov Chain for an M/M/1 Queue with State-Dependent Service Time Distribution . . . . . 21

Figure 3.4. Markov Chain for the Mass Join Operation . . . . . 23

Figure 3.5. Overview of Performance Model with Leave Operation . . . . . 28

Figure 3.6. An illustration for the change in the inter-arrival time distribution due to the Join Flood Attack Model . . . . . 31

Figure 3.7. An illustration for the change in the residence time distribution due to the Immediate Leaving Attack Model . . . . . 32

Figure 3.8. An illustration for the change in the residence time distribution due to the Disconnector Attack Model . . . . . 33

Figure 4.1. An overview of join-PBC operation in KAP-PBC . . . . . 40

Figure 4.2. Graph of Average Waiting Time in KAP-PBC against Participant Arrival Rate for different Mean Residence Time . . . . . 46

Figure 5.1. An Illustration of File Re-Encryption Process in PFSS . . . . . 50

Figure 5.2. Representation of the Average Waiting Time in PFSS with respect to Participant Arrival Rate for different Mean Residence Time . . 57

Figure 5.3. Representation of the Average Waiting Time in PFSS with respect to Participant Arrival Rate for different Internet Connection Bandwidth . . . . . 58

# LIST OF TABLES

Table 2.1. Performance evaluation methods used in well-known group key agreement protocols . . . . . 15

Table 2.2. Performance evaluation methods used in well-known secure file sharing systems . . . . . 16

Table 3.1. Definition of the Parameters used in the Residence Model . . . . . 18

Table 3.2. Definition of the Parameters used in the Join Model . . . . . 20

Table 3.3. Definition of the Parameters used in the Leave Model . . . . . 27

Table 4.1. KAP-PBC functions and their notations . . . . . 35

Table 4.2. KAP-PBC functions and their execution times on a computer with 1.8 GHz Intel Core i5 processor, for AES-256 (CBC mode) as the cipher . . . . . 45

# LIST OF SYMBOLS

| | |
|---|---|
| $\mathcal{B}$ | Internet connection bandwidth of participants |
| $E[S_x]$ | The mean service time of operation $x$ |
| $E[S_{join}^k]$ | The mean service time of join operation when there are $k$ group participants |
| $E[Y]$ | The mean number of joining participants in mass join operation |
| $\mathcal{F}$ | Size of a transmitted file |
| $k$ | The number of group participants |
| $L$ | The average number of participants who are waiting to join |
| $L_q$ | The average number of participants in the queue who are waiting to join |
| $P_k$ | The probability that there are $k$ participants in the group |
| $t_F$ | File transmission time |
| $t_{enc+dec}$ | Time required to encrypt and decrypt a file in PFSS |
| $t_{gkc}$ | Time required to perform Group Key Computation function in KAP-PBC |
| $t_{pkb}$ | Time required to perform Temporary Public Key Broadcast function in KAP-PBC |
| $t_{pkc}$ | Time required to perform Temporary Public Key Computation function in KAP-PBC |
| $t_{pkv}$ | Time required to perform Temporary Public Key Verification function in KAP-PBC |
| $t_{skb}$ | Time required to perform Session Key Broadcast function in KAP-PBC |
| $t_{skc}$ | Time required to perform Session Key Computation function in KAP-PBC |
| $t_{skv}$ | Time required to perform Session Key Verification function in KAP-PBC |
| $t_{upl+dow}$ | Time required to upload and download a file in PFSS |
| $W$ | The average waiting time of a joining participant |
| $W_q$ | The average waiting time spent in the queue by a joining participant |

| | |
|---|---|
| $\lambda$ | Arrival rate of new participants |
| $\mu$ | Residence rate of group participants |
| $\mu_x$ | Service rate of operation $x$ |
| $\rho$ | Server utilization rate |
| $\sigma_a^2$ | Variance of the participant arrival distribution |
| $\sigma_s^2$ | Variance of the service time distribution |
| $\tau_1$ | A symbol that is used to represent the following expression: $t_{pkv} + 2t_{skv} + t_{skc} + t_{pkc} + t_{gkc} - t_{skb} - t_{pkb}$ |
| $\tau_2$ | A symbol that is used to represent the following expression: $2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc}$ |
| $\tau_3$ | A symbol that is used to represent the following expression: $t_{pkv} + t_{skv} + t_{pkc} + t_{skc} - 2t_{pkb} - 2t_{skb} + t_{gkc}$ |

# LIST OF ACRONYMS/ABBREVIATIONS

| | |
|---|---|
| DDoS | Distributed Denial of Service |
| DGKAP | Dynamic Group Key Agreement Protocol |
| GKAP | Group Key Agreement Protocol |
| KAP-PBC | Key Agreement Protocol with Partial Backward Confidentiality |
| PFSS | Private File Sharing System |
| SFSS | Secure File Sharing System |

# 1. INTRODUCTION

Group key agreement protocols (GKAPs) are instrumental to establish a secure communication for a set of participants. In such protocols, the security of a communication is accomplished via using a common key computed by all cooperating participants in the group. Therefore, a central authority is not required in GKAPs. Diffie-Hellman Key Exchange Protocol [1] is the first key agreement protocol that operates for a group of two participants. Then, the idea was extended to multiple participants by Ingemarsson *et al.* in [2]. Following the multiple participant setup, two important group key agreement protocols with and without authentication were proposed by Burmester and Desmedt in [3]. Authentication property is used for confirming the identities of participants in group communications. Another significant security property is the fault-tolerance property [4], which is used for detecting and removing the malicious participant from the group key computation. Forward secrecy property [5] also plays an important role for the security of group key agreement protocols. The property was adopted by Tseng [6] for protecting against the compromise of former and subsequent group keys of a protocol, if the long-term key of a participant is compromised.

Former GKAPs mostly operate on static groups [7, 8, 9], in which the set of participants remains unchanged till the end of a communication session or the protocol is re-executed from the beginning when the set of participants is altered. However, recent trends in communication technologies require dynamic settings for the set of participants due to the significant overhead of updating the group key. Accordingly, group key agreement protocols have evolved to overcome such overhead by providing dynamic group operations [10, 11, 12]. The most common dynamic group operations are the join of new participants into the group and the leave of existing participants from the group. Such join and leave operations are sometimes called as auxiliary group key agreement operations. In contrast to static group key agreement protocols, dynamic group key agreement protocols (DGKAPs) support auxiliary operations by selecting a small subset of participants as active participants. Active participants, which are entities in the group, are responsible for updating the group key by re-

executing the protocol from the beginning after any join or leave has occurred. Since the set of active participants is a smaller group than the original set, DGKAPs provide better performance than static ones.

## 1.1. Motivation

Performance analysis of DGKAPs is a significant issue for evaluating the applicability of a protocol in a real-life application. However, performance analysis of well-known DGKAPs only consider numerical methods, simulation techniques and asymptotic analysis, and to the best of our knowledge, there is no formal performance model for scalability analysis of DGKAPs. Therefore, it is not possible to measure effects of auxiliary operations on the group communication. For instance, a set of malicious participants may simultaneously send frequent join requests to a communication group, as part of a Distributed Denial of Service (DDoS) attack. If there exist frequent join requests to the group then, group members spend most of their time to update the group key rather than communicating with each other. Consequently, such frequent requests may violate the availability of the group communication. Provision of a formal scalability analysis by employing queueing models is expected to help learning availability boundaries of the group communication, which is our main motivation for this thesis. Learning availability boundaries of a DGKAP can help to detect DDoS attacks and provide the ability to take counter measurements. In addition, by using formal models, we aim to address the following issues: (i) the average waiting time of a joining participant before joining the group, (ii) the effects of frequent participant arrivals on the availability of group communication, (iii) the interdependence between the average waiting time of a joining participant and the number of participants in the group.

## 1.2. Contributions

In this thesis, we propose the following contributions to solve the scalability issue of DGKAPs:

- We propose a formal performance model on DGKAPs by using queueing models to analyze the scalability of protocols. Our performance model can be used for detecting the conditions where the system availability is violated.
- We illustrate the applicability of our model on key agreement protocol with partial backward confidentiality (KAP-PBC) [11]. We conduct a theoretical analysis on KAP-PBC and present the numerical performance results.
- We also design attack models that can be used against our performance model. By using these attack models, we display how the targeted system is affected during an attack.
- Moreover, we extend our performance model for secure file sharing systems that utilize group key agreement protocols by modelling file download/upload and file encryption/decryption operations.
- Furthermore, we design a healthcare scenario and present a demonstrative use case of our model on Private File Sharing System (PFSS) [11]. We present numerical performance results of the system based on the designed scenario.

### 1.3. Thesis Outline

The rest of the thesis is organized as follows:

In Chapter 2, we present a literature overview of group key agreement protocols, secure file sharing systems and queueing theory. In addition, we provide a comparison of performance analysis methods used in secure multi-party communication systems.

In Chapter 3, we explain the details of the proposed queueing-based performance model. Moreover, we introduce possible attacking models on the proposed performance model.

In Chapter 4, an application of the proposed model on Key Agreement Protocol with Partial Backward Confidentiality (KAP-PBC) [19] is given.

In Chapter 5, we extend the proposed performance model for secure file sharing systems and design a healthcare scenario as a demonstrative use case of our model. In addition, we present an application of the model on Private File Sharing System [11].

Finally, in Chapter 6, we conclude the thesis and explain our future research directions.

# 2. MATHEMATICAL BACKGROUND AND LITERATURE OVERVIEW

In this chapter, we present a detailed overview for the thesis regarding the queueing theory and multi-party group communication systems. First, we present the mathematical background for the queueing theory. Then, we review multi-party group communication systems with respect to dynamic group key agreement protocols (DGKAPs) and secure file sharing systems (SFSSs). Finally, we present a detailed comparison for the performance analysis approaches of DGKAPs and SFSSs.

## 2.1. Mathematical Background and Definitions for Queueing Theory

Queueing theory is a way to mathematically predict the queue lengths and waiting time in various application areas such as telecommunication, traffic engineering, industrial engineering, etc. A queue can be defined as a line in which entities are waiting to be processed [13] and it is formally represented by using the Kendall's Notation:

**Definition 2.1** *Let queue Q has A/B/C/D/E Kendall's Notation. Each Kendall's Notation symbol of Q is explained as follows:*

- *A: The distribution of customer arrival time. There are three possible options:*
  - *(i) A=D: Arrival time of customers are pre-determined.*
  - *(ii) A=M: Arrival time of customers is a Poisson distributed random variable. In other words, inter-arrival time of customers is an exponentially distributed random variable.*
  - *(iii) A=G: Arrival time has a general distribution with a specific mean and variance.*
- *B: The distribution of customer service time. There are three possible options:*
  - *(i) B=D: Service time of customers are deterministic.*
  - *(ii) B=M: Service time of customers is an exponentially distributed random vari-*

*able.*

*(iii) B=G: Service time has a general distribution with a specific mean and variance.*

- *C: The number of servers which work simultaneously. It is assumed that one server can only serve one customer at a time.*
- *D: The capacity of the queue. D represents how many customers can wait in the queue. If a customer arrives when the queue capacity is full, the customer will leave without entering the queue. If D is omitted in the notation, it is assumed that the queue capacity is infinite.*
- *E: The capacity of the customer population. E represents how many people at most can arrive into the system. If E is omitted in the notation, it is assumed that the customer population is infinite and there is no limit for the customer who can arrive into the system.*

In addition to the Kendall's notation, a queue also has input parameters such as the mean arrival rate and the mean servic rate. These input parameters affect the results of analysis of the queueing model as defined in [13] and the formal definition is as given in Definition 2.2:

**Definition 2.2** *Let Q be a X/Y/Z queue. The list of input parameters of Q are given as follows:*

- *Mean Arrival Rate ($\lambda$): The mean arrival rate is a parameter which states how many customers, on average, arrives into the system per unit time.*
- *Mean Service Rate ($\mu$): The mean service rate is a parameter which represents the mean customer number that are served in unit time. In other words, on average $\mu$ number of customers are served in a unit time.*
- *Mean Service Time ($E[S]$): The mean service time represents how much time on average a customer spends in the server. The mean service time is the inverse of*

*the mean service rate and can be expressed as follows:*

$$E[S] = \frac{1}{\mu}$$

- *Server Utilization ($\rho$): The server utilization represents the fraction of time in which a server is busy. The server utilization can be expressed as:*

$$\rho = \frac{\lambda}{C\mu}$$

- *Arrival Rate Variance ($\sigma_a^2$): The variance of the customer arrival distribution. The arrival rate variance is zero if arrivals are deterministic:*

$$X = D \Rightarrow \sigma_a^2 = 0$$

- *Service Time Variance ($\sigma_s^2$): The variance of the service time distribution. If service durations are deterministic, service time variance is zero:*

$$Y = D \Rightarrow \sigma_s^2 = 0$$

As the first study regarding queeuing theory, Erlang designed a model for number of arrivals at the Copenhagen telephone exchange. He considered Poisson arrivals and deterministic service time with a single server. The model in the study is *M/D/1* queue by Kendall's notation. Later, Erlang extended his model for multiple servers as M/D/k queue in [14].

Queueing Theory, after Erlang's studies, attracted more attention by researchers. Pollaczek considered a model where number of arrivals is Poisson distributed but the service time is not deterministic. In his model, the service time has an arbitrary distribution. This model is known as *M/G/1* queue by Kendall's notation. Pollaczek proposed a formal model for the performance metrics of an M/G/1 queue by using integral equations. Later, Khincine derived the same model by using probability theory.

Therefore, the formula for the performance metrics of an M/G/1 queue is called as Pollaczek-Khincine formula and defined as follows:

$$W = \frac{\rho + \lambda\mu\sigma_s^2}{2(\mu - \lambda)} + \frac{1}{\mu}$$

$$L = \frac{\rho^2 + \lambda^2\sigma_s^2}{2(1 - \rho)} + \rho$$

where $W$ is the mean customer waiting time and $L$ is the mean number of customers. Other parameters are as given in Definition 2.2.

The main performance metrics used in queueing theory are the average queue length and the average waiting time. The list of widely used performance metrics are given as follows:

**Definition 2.3** *Let $Q$ be a X/Y/Z queue. The list of performance metrics of $Q$ are given as follows:*

- *Mean Customer Waiting Time ($W$): In the long run of a queueing simulation, the mean customer waiting time represents how much time, on average, a customer has to wait in the system before the customer is served. $W$ includes both the time spent in the queue and the time spent in the server.*
- *Mean Customer Waiting Time in Queue ($W_q$): In the long run of a queueing simulation, $W_q$ represents how much time, on average, a customer has to wait in the queue before the customer is served. The relation between $W$ and $W_q$ can be expressed as follows:*

$$W = W_q + E[S]$$

- *Mean Number of Customers (L): The mean number of customers represents how many customers are expected to be in the system in the long run of a simulation. L includes both the number of customers in the system and the number of customers in the queue.*
- *Mean Number of Customers in Queue ($L_q$): The mean number of customers in queue represents how many customers are expected to be in the queue in the long run of a simulation. The relation between L and $L_q$ can be expressed as follows:*

$$L = L_q + \lambda E[S]$$

In many applications, M/M/1 queue model or M/G/1 queue model are used for the sake of simplicity. However, these queue models require the assumption that arrivals are Poisson distributed. Such assumption may not hold for complicated systems. Therefore, a more general queueing model, *G/G/1* was suggested [15] for such systems. For performance metrics such as the average waiting time, an approximated solution was provided by Kingman in [16].

In addition to the aforementioned studies, state-dependent queues were also proposed and analyzed. State-dependent queues indicate that either arrival distribution or service time distribution, or both, changes over time. Such change can be triggered by parameters like the average queue length or the average waiting time. For instance, Bekker *et al.* proposed two different models in which service time is affected by the waiting time of the first customer in queue [17]. The first model has a single server whereas the second model has a secondary server that acts a suplementary server. Abouee-Mehrizi and Baron proposed an M/G/1 queueing model in which both arrival rates and service times depend on the number of customers in the system [18]. Authors denoted their model as $M_n/G_n/1$ and provided steady-state equations for number of customers in the system. However, authors specified that Little's Law[19] cannot be used in $M_n/G_n/1$ since both arrival rate and service rate are affected by future arrivals. As a consequence, retrieving sojourn time distribution becomes a hard problem.

Another study regarding state-dependent queueing models was conducted by Banik in [20]. In the study, a queueing model denoted as *GI/BMSP/1* was analyzed. GI indicates that arrivals are generally distributed and independent from each other. BMSP, abbreviation for Batch Markovian Service Process, represents a system where services are completed in batches and Markovian. Banik proposed a *GI/BMSP/1* queue where inter-arrival times depend on the queue length and provide steady-state equations for the model. Laghaie *et al.* proposed a *D/M/1* queue with deteriorating server [21]. In other words, authors suggested a queueing model with deterministic arrivals and exponential service times where the service time increases as more customers are served. The model was built to represent a production line where the server is subject to gradual deterioration. Both constant rate deterioration and stochastic deterioration were considered, in separate models. Moreover, authors suggested a maintenance policy for minimizing deterioration in such systems.

## 2.2. Group Key Agreement Protocols

Early studies of group key agreement protocols focused on static groups. In [2], Inglamersson *et al.* proposed a conference key distribution system where a common key is generated for all participants in the group. Based on this system, Wu proposed user anonymity property [22] for preventing identities of participants from each other. Therefore, based on the anonymity definition given in [22], only the organizer of the conference communication is able to know identities of the attended participants. Then, Burmester and Desmedt proposed two group key agreement protocols, with and without an authentication mechanism. In group key agreement protocols, authentication property is used for confirming the identities of participants in group communications. Tzeng adopted fault-tolerance property for group key agreement protocols in [4]. Fault-tolerance property is used for detecting and removing the malicious participant from the group key computation. Chang *et al.* worked on Tzeng's protocol and improved the performance of group key computation [23]. Wu *et al.* proposed an asymmetric group key agreement protocol where one common key is used for encryption, but each participant computes and uses a unique decryption key [24]. Tan *et al.* proposed an asymmetric group key agreement protocol for mobile cloud environments in [25].

The need for dynamic groups emerged as communication technologies evolved and the number of participants in group communications increased. Dynamic group key agreement protocols (DGKAPs) reduce the computational and communicational overhead when the set of participants change. One of the well-known examples of DGKAPs was presented by Tzeng in [4]. The protocol provides fault-tolerance property which enables the detection and correction of malicious participant actions during the group key computation. Chuang and Tseng proposed an efficient DGKAP in [26]. The protocol has authentication property and was designed for low-power mobile devices. Bresson *et al.* proposed a DGKAP for low-power mobile devices in [27]. Xu *et al.* proposed a DGKAP for vehicular ad hoc networks in [28]. Qikun *et al.* proposed an asymmetric and dynamic group key agreement protocol for telemedicine communication systems in [29].

There are two more significant security properties for group key agreement protocols, namely backward confidentiality and forward confidentiality[30]. In backward confidentiality, participants who join the group cannot compute the former group keys. In forward confidentiality, participants who leave the group cannot compute the subsequent group keys. Tseng proposed a group key agreement protocol which has both backward confidentiality and forward confidentiality properties in [10]. Lee *et al.* realized the security weakness in Tseng's second protocol and proposed an attack model along with a possible countermeasure in [31]. Protocols proposed by Cheng *et al.* [12] and Ermis *et al.* [30] are another examples for DGKAPs and both protocols provide fault-tolerance, backward confidentiality and forward confidentiality properties. Moreover, computational cost and transmission cost of these protocols are better than the existing ones.

### 2.3. Secure File Sharing Systems

A secure file sharing system (SFSS) is s system that registered participants are able to securely share their files with other registered participants. Registered participants use a common group key to encrypt their files before uploading them to the storage of SFSS. Other registered participants can access the content of shared files

by downloading them and decrypting them with the group key. Since shared files are stored encrypted in a SFSS, unregistered participants cannot access the content of these files. In Many SFSSs, cloud technology is utilized as storage service. Most well-known examples of such services are Amazon S3[32], Microsoft OneDrive[33], Google Drive[34], Apple iCloud[35] and Dropbox[36].

The first secure file sharing systems, namely Enclaves, were designed by Gong in 1997 [37]. Enclaves run on top of TCP/IP layer and do not require super user privileges. A group manager is responsible for controlling whether a file is modified or not. MONA proposed by Liu *et al.* in [38] was designed for handling dynamic group operations in an efficient manner. A group manager is responsible to register new participants and trace the identity of a malicious participant if a suspicious action is performed. There are other similar schemes, which require a fully trusted group manager, such as systems proposed in [39] and [40]. Although having a fully trusted group manager provides easier management, it also endangers the availability due to the single-point-of-failure property of the system. Sharan and Shrikanth proposed a scheme where key-aggregation is used [41]. In their scheme, file uploader uses a different encryption key for each of the uplaoded file. When another user downloads a subset of the uploaded files, decryption keys of the relevant files are aggregated and sent to the downloading user in a secure manner. Xiong *et al.* proposed an end-to-end content protection scheme called CloudSeal [42], which is specifically designed to secure data in content delivery networks [43]. CloudSeal provides an efficient and secure data storage in such networks.

Some researchers adopted overlay approach for SFSS design. In overlay approach, there is another layer between the Cloud storage and the local file systems of participants. Duarte *et al.* proposed a system namely Protbox, in which there is an overlay infrastructure between cloud folders and the local replications of the corresponding cloud folders [44]. The overlay folders are called Prot folders. Each Prot folder is associated with a cloud folder. This association is referred as Protbox pair and has a unique pair key. Tang *et al.* proposed a SFSS called FADE [45]. FADE has an intermediate layer which provides policy-based access control and policy-based file deletion. Yamai

*et al.* proposed another overlay-based system for sharing files on different administrative domains [46]. In their system, a proxy computer is used as the intermediate layer between the clients and servers.

Yu *et al.* proposed a SFSS [47] which provides fine-grained access control and data security properties by utilizing ciphertext-policy attribute-based encryption [48]. The files are stored in semi-trustable proxy servers. Such servers perform all the tasks correctly but try to learn the content of the stored files. The system uses proxy re-encryption [49] to reduce the computational overhead on the participants by delegating the computationally heavy tasks to the proxy servers. However, the study does not cover a performance analysis.

Wang *et al.* proposed a secure file sharing scheme named HABE that provides fine-grained access control and participant removal with high performance[50]. HABE combines the concept of hierarchical identity-based encryption [51] and ciphertext-policy attribute-based encryption [52]. In HABE, proxy re-encryption and lazy revocation [53] are used for improving performance of participant removal. However, for the performance analysis, only the internal trusted party is considered. In addition, the performance of the system under frequent participant joins or leaves is not analyzed. The same authors proposed another scheme called TimePRE [54] which is an extension of HABE. In TimePRE, the proxy re-encryption process is automated into pre-defined time slots to improve the performance of user removals. The performance of TimePRE is analyzed more extensively compared to HABE. Nevertheless, a formal model based on the frequency of participant joins or leaves is not available.

Ermis *et al.* propose Private File Sharing System (PFSS) that provides participant revocation mechanism and fine-grained access control [11]. PFSS utilizes Key Agreement Protocol with Partial Backward Confidentialty (KAP-PBC) which provides backward confidentiality and forward confidentiality. In addition, a new security property named partial backward confidentiality is introduced in the study. In partial backward confidentiality, only a single joining participant can compute the last valid group key just before joining the group. Other participants cannot compute any former

group key. The performance of PFSS is measured by using asymptotic analysis and simulation. However, a formal performance model is not designed for PFSS.

## 2.4. A Comparison of Performance Evaluation Methods used in Secure Multi-Party Group Communication Systems

In this section, we present a comparison of performance evaluation methods used in different secure multi-party group communication systems. First, we focus on the performance evaluation methods presented in DGKAPs. Then, we compare secure file sharing systems by using performance evaluation methods. While comparing DGKAPs and secure file sharing systems, we use the following comparison criteria:

- Asymptotic Complexity: We consider asymptotic complexity analysis as a derivation of an upper bound regarding time complexity of a system by using $\mathcal{O}$ notation.
- Numerical Methods: We consider numerical methods as a subset of asymptotic complexity analysis. Numerical analysis is performed when time complexity of the system is estimated for a fixed number of participants but not derived asymptotically by using $\mathcal{O}$ notation.
- Simulation: Performance analysis is performed via simulation when the system runs on a simulation environment for different ranges of parameters.
- Formal Performance Model: We consider networks of queues as formal performance models. If the performance of a communication system is analyzed by using a queueing model, we state that formal performance models are used.

Methods used for performance evaluation of well-known DGKAPs are as given in Table 2.1. Comparisons show that numerical methods are the most widely used approach to analyze the performance of protocols. Simulation is the second most common method and asymptotic complexity analysis is less widely utilized. Only three protocols [23, 30, 59] use both simulation and asymptotic complexity analysis simultaneously. However, none of the example DGKAPs were analyzed by using formal performance models. The study in [64] compares four different DGKAPs with respect

Table 2.1. Performance evaluation methods used in well-known group key agreement protocols

| | **Numerical Methods** | **Asymptotic Complexity** | **Simulation** | **Formal Performance Model** |
|---|---|---|---|---|
| Abdel-Harfez *et al.*[55] | ✓ | ✗ | ✓ | ✗ |
| Augot *et al.*[56] | ✗ | ✗ | ✓ | ✗ |
| Change *et al.*[57] | ✓ | ✗ | ✓ | ✗ |
| Chang *et al.*[23] | ✓ | ✓ | ✓ | ✗ |
| Cheng *et al.*[12] | ✓ | ✗ | ✓ | ✗ |
| Chung[58] | ✓ | ✗ | ✗ | ✗ |
| KAP-PBC[30] | ✓ | ✓ | ✓ | ✗ |
| GKAP-MANET [59] | ✓ | ✓ | ✓ | ✗ |
| Gangwar *et al.* [60] | ✓ | ✓ | ✗ | ✗ |
| Huang *et al.*[9] | ✓ | ✓ | ✗ | ✗ |
| Li *et al.*[61] | ✓ | ✗ | ✓ | ✗ |
| Teng *et al.*[62] | ✓ | ✗ | ✗ | ✗ |
| Tzeng[8] | ✓ | ✗ | ✗ | ✗ |
| Tzeng[4] | ✓ | ✗ | ✗ | ✗ |
| Zhang *et al.*[63] | ✓ | ✓ | ✗ | ✗ |
| Zhao *et al.*[7] | ✗ | ✗ | ✓ | ✗ |

to scalability of protocols, however; it does not provide a formal model.

DGKAPs are also utilized in secure file sharing systems as a file confidentiality service by using the computed group key to encrypt and decrypt shared files. Methods used for performance analysis of some well-known SFSSs are as given in Table 2.2. According to the comparison, all SFSSs use numerical methods for evaluating their performance. HABE[50], which is a combination of hierarchical identity-based encryption [51] and ciphertext-policy attribute-based encryption [52], uses proxy re-encryption

Table 2.2. Performance evaluation methods used in well-known secure file sharing systems

| | **Numerical Methods** | **Asymptotic Complexity** | **Simulation** | **Formal Performance Model** |
|---|---|---|---|---|
| System in [65] | ✓ | ✓ | ✓ | ✗ |
| PFSS[11] | ✓ | ✓ | ✓ | ✗ |
| System in [66] | ✓ | ✓ | ✗ | ✗ |
| mCP-ABE [67] | ✓ | ✗ | ✗ | ✗ |
| TimePRE[54] | ✓ | ✓ | ✓ | ✗ |
| Mona[38] | ✓ | ✓ | ✓ | ✗ |
| System in [39] | ✓ | ✗ | ✓ | ✗ |
| FADE[45] | ✓ | ✗ | ✗ | ✗ |
| HABE[50] | ✓ | ✓ | ✗ | ✗ |
| System in [47] | ✓ | ✗ | ✗ | ✗ |
| System in [40] | ✓ | ✗ | ✓ | ✗ |

[49] and lazy revocation [53] to improve performance of participant removal operation. However, the study does not provide simulation results or formal performance models. None of the example SFSSs presented in Table 2.2 use formal performance models.

# 3. ANALYTICAL MODELS FOR THE SCALABILITY ANALYSIS OF DYNAMIC GROUP KEY AGREEMENT PROTOCOLS

In this chapter, we introduce a formal performance model for the scalability analyis of dynamic group key agreement protocols (DGKAPs) by using queueing models. We provide details about our approach and explain how we model operations, namely single join, mass join and leave. We also propose possible scenarios where our performance model can be used to analyze the scalability of a DGKAP. Moreover, we design attack models that can be used by malicious participants in our designed scenarios.

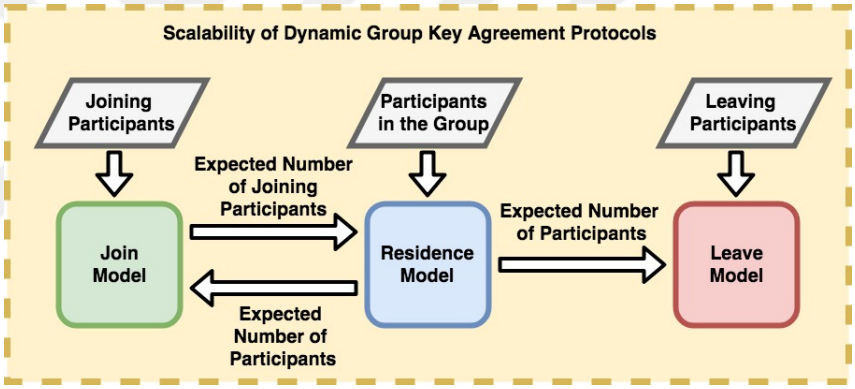



Figure 3.1. An Overall View of the Proposed Performance Model for Dynamic Group Key Agreement Protocols

An overall view of the proposed performance model is as shown in Figure 3.1. First, we model the number of participants in the group by using the residence model. Then, we propose analytical models for single and mass join operations by using the expected number of participants in the group. Finally, we propose a model for the leave operation with respect to the expected number of participants and the set of leaving participants. Consequently, we are able to obtain analytical results for the average waiting time and the average number of waiting participants for single join, mass join and leave operations of DGKAPs.

### 3.1. Residence Model

The group size significantly affects the time required to perform the group key computation. We consider the group size as a parameter which is represented with an infinite server queueing model, namely Residence Model. Parameters used in Residence Model is as given in Table 3.1.

Table 3.1. Definition of the Parameters used in the Residence Model

| **Parameter** | **Explanation** |
|---|---|
| $\lambda$ | The mean arrival rate of new participants |
| $\mu$ | The mean group residence rate |
| $k$ | The number of existing participants in the group |
| $P_k$ | The probability that there are $k$ existing participants in the group |
| $s_b$ | The number of busy servers in Residence Model |

In Residence Model, the service time is considered as the residence time of a participant in the group until participants leave the group. Each participant in the group is served by a different server. Hence, the number of busy servers at time $t$, equals to the number of participants in the group at time $t$:

$$s_b(t) = k(t) \tag{3.1}$$

where $s_b$ represents the number of busy servers and $k$ represents the number of participants in the group.

We assume that inter-arrival time in the residence time model is exponentially distributed and the residence time of participants is arbitrarily distributed. Therefore, we use an $M/G/\infty$ queue. We derive the performance metrics of an $M/G/\infty$ queue as shown in [68]:

$$P_1 = \frac{\lambda}{\mu} P_0 \tag{3.2}$$

$$P_k = \frac{\lambda}{k\mu} P_{k-1} \tag{3.3}$$

where $k$ is the number of participants in the group, $\lambda$ is the arrival rate, $\mu$ is the service rate and $P_k$ is the steady-state probability. By combining Equation 3.2 and 3.3, we obtain

$$P_k = \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!} P_0 \tag{3.4}$$

Since $\sum_{k=0}^{\infty} P_k = 1$, we also derive $P_0$ in terms of $\lambda$ and $\mu$ by using Taylor's Expansion:

$$P_0 \left(1 + \frac{\lambda}{\mu} + \frac{\left(\frac{\lambda}{\mu}\right)^2}{2!} + \frac{\left(\frac{\lambda}{\mu}\right)^3}{3!} + ...\right) = 1 \tag{3.5}$$

$$P_0 e^{\frac{\lambda}{\mu}} = 1 \tag{3.6}$$

$$P_0 = e^{-\frac{\lambda}{\mu}} \tag{3.7}$$

By substituting $P_0$ in Equation 3.7 into Equation 3.4, we derive the closed-form expression for $P_k$ as

$$P_k = \frac{\left(\frac{\lambda}{\mu}\right)^k e^{-\frac{\lambda}{\mu}}}{k!} \tag{3.8}$$

We observe that Equation 3.8 is the probability mass function of the Poisson distribution with mean $\lambda/\mu$. Therefore, the group size parameter is considered as a Poisson-distributed random variable.

## 3.2. Join Model

In Join Model, we analyze the scalability of single join and mass join operations. First, we focus on single join operation where a participant joins the group at a time. Then, we focus on mass join operation where multiple participants join the group simultaneously. Parameters used in Join Model are as given in Table 3.2.

Table 3.2. Definition of the Parameters used in the Join Model

| **Parameter** | **Explanation** |
|---|---|
| $\lambda$ | The mean arrival rate of new participants |
| $\mu_{join}$ | The mean joining participant rate |
| $\mu$ | The mean group residence rate |
| $S_{join}^{k}$ | The service time for a new participant to join the group when there are $k$ participants in the group |
| $W$ | The average waiting time of a new participant before joining |
| $L$ | The average number of participants waiting for join operation at an arbitrary time |
| $k$ | The number of existing participants in the group |
| $P_k$ | The probability that there are $k$ existing participants in the group |

### 3.2.1. Single Join Operation

We model the single join operation by using an M/M/1 queue. When a participant requests to join the group, the request is considered as an arrival event into the queue. The completion of a single join operation is considered as a departure event from the queue. We assume that the inter-arrival time for a single join operation is exponentially distributed. We also assume that the service time is exponentially distributed for simplicity. Therefore, for the single join operation, we use a queue which has a single server, exponential inter-arrival time and exponential service time.

The overview of the model used for the join operation is as shown in Figure 3.2. The left component of the figure represents the queueing model for the arriving join requests. When new participants arrive, they wait in the queue until they join the group by using the join operation. Then, they arrive at the second queueing model on the right, which represents the queueing model for the group sizes. This is an infinite server model in which participants do not wait in the queue. When a participant decides to leave the group, the service is considered completed and the participant leaves. The service time represents how much time a participant stays in the group. In

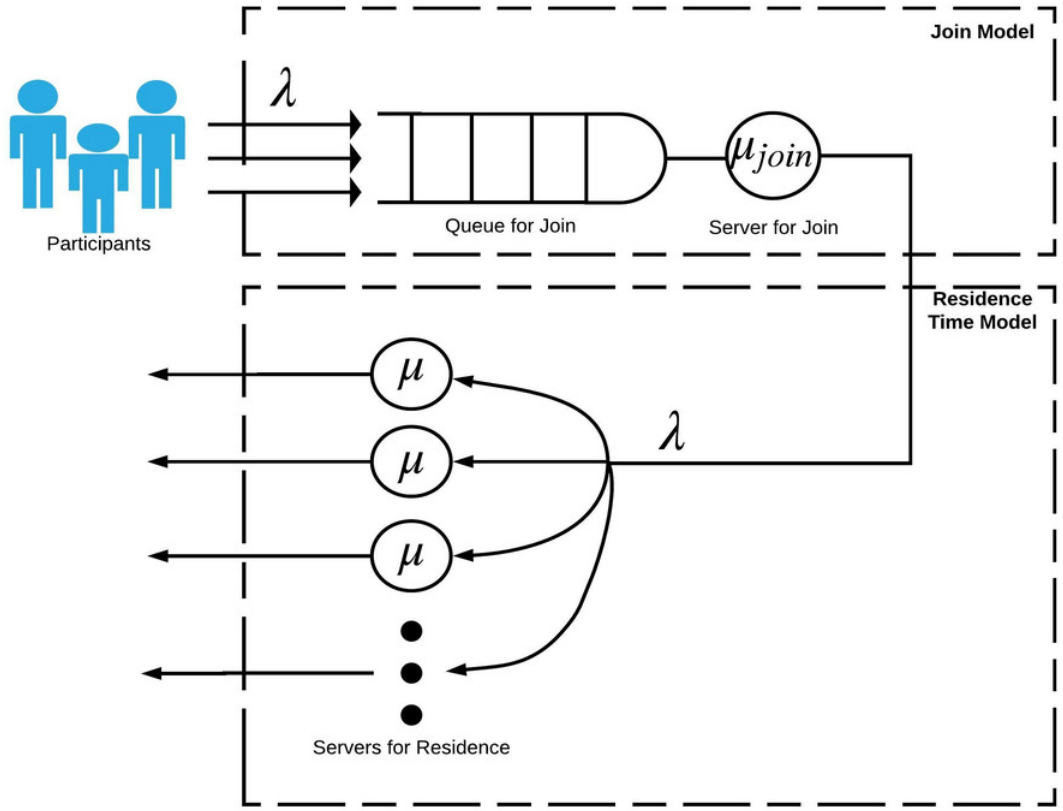



Figure 3.2. The model for the single join operation

this model, the number of busy servers corresponds to the number of participants in the group. We can obtain information about the group size distribution by using this model.

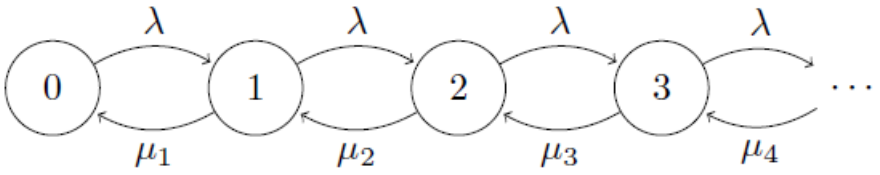



Figure 3.3. Markov Chain for an M/M/1 Queue with State-Dependent Service Time Distribution

In Join Model, we use an M/M/1 queue with state-dependent mean service time[13] where the service time is determined by the number of participants in the group. An M/M/1 queue with state-dependent service time distribution can be represented by the Markov Chain shown in Figure 3.3. $\lambda$ denotes the arrival rate and $\mu_{join}$

denotes the service rate.

In Equation 3.8, we have deduced that the group size, denoted as $k$, is Poisson-distributed with rate $\frac{\lambda}{\mu}$. Thus, the expected value of $k$ equals to

$$\begin{aligned} E[k] &= E\left[Poisson\left(\frac{\lambda}{\mu}\right)\right] \\ E[k] &= \frac{\lambda}{\mu} \end{aligned} \tag{3.9}$$

The performance metrics of M/M/1 queues are derived in [69] as follows:

$$W = \frac{1}{\mu - \lambda} \tag{3.10}$$

$$L = \frac{\lambda}{\mu - \lambda} \tag{3.11}$$

Using the result in Equation 3.9, we approximate the performance metrics by fixing the group size to $\lambda/\mu$. As a result, the system becomes a regular M/M/1 queue with constant mean service time. Then, we define the approximated mean service time as follows:

$$E[S_{join}^{apr}] = \begin{cases} E\left[S_{join}^{\frac{\lambda}{\mu}}\right] & \frac{\lambda}{\mu} \in \mathcal{Z} \\ \dfrac{E\left[S_{join}^{\lceil \frac{\lambda}{\mu} \rceil}\right] + E\left[S_{join}^{\lfloor \frac{\lambda}{\mu} \rfloor}\right]}{2} & \frac{\lambda}{\mu} \notin \mathcal{Z} \end{cases}$$

Using the value of $E[S_{join}^{apr}]$ in performance metric equations yields the final results:

$$W = \frac{1}{\frac{1}{E\left[S_{join}^{apr}\right]} - \lambda} \tag{3.12}$$

$$L = \frac{\lambda}{\frac{1}{E\left[S_{join}^{apr}\right]} - \lambda} \tag{3.13}$$

### 3.2.2. Mass Join

Mass join operation allows multiple participants to simultaneously join the group such as protocols in [30, 11]. During the modeling of the mass join operation, we use batch queues [70]. We assume that both inter-arrival time and service time of the mass join operation is exponentially distributed. We also assume that new participants arrive one by one but join in bulks. Therefore, we use a single server queue with batch service to represent the mass join operation.

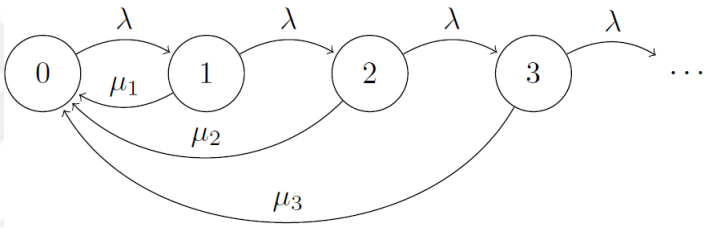



Figure 3.4. Markov Chain for the Mass Join Operation

The batch size of the join service, $Y$, is not constant since it depends on the number of participants waiting in the queue to join the group. In general, $Y$ in bulk service queues is either fixed or considered as a random variable which is smaller than the queue size [71]. However, we assume that the system always allows all participants in the queue to join the group. Therefore, $Y$ is equal to the number of participants waiting in the queue.

Figure 3.4 shows the Markov chain for calculating the steady-state probabilities in the mass join operation. Since the batch size is considered the same as the queue size, each state goes back to the empty state after a departure event. In the mass join operation, the number of joining participants may affect the service time of the operation. Each state has its own service time denoted as $\mu_1$, $\mu_2$... We derive the

balance equations from the Markov chain as

$$P_k\lambda = P_{k+1}(\lambda + \mu_{k+1}) \tag{3.14}$$

which equals to

$$P_k = \left(\frac{\lambda^k}{\prod_{i=1}^{k}(\lambda + \mu_i)}\right) P_0 \tag{3.15}$$

$$\tag{3.16}$$

By substituting Equation 3.15 into $\sum_{k=0}^{\infty} P_k = 1$, we obtain

$$P_0 + P_0 \sum_{k=1}^{\infty} \left(\frac{\lambda^k}{\prod_{i=1}^{k}(\lambda + \mu_i)}\right) = 1 \tag{3.17}$$

$$\tag{3.18}$$

When we leave $P_0$ on a side of the equation, we obtain the closed-form expression of $P_0$ as

$$P_0 = \left[1 + \sum_{k=1}^{\infty} \left(\frac{\lambda^k}{\prod_{i=1}^{k}(\lambda + \mu_i)}\right)\right]^{-1} \tag{3.19}$$

The closed-form of $P_0$ given in Equation 3.19 can be applied to any dynamic group key agreement protocol that supports the mass join operation. In some dynamic group key agreement protocols, the number of joining participants in the mass join operation has a negligible effect on the group key update time. For such protocols, the following condition holds:

$$\mu_i = \mu \;\; for\, i \in \{1, 2, ...\} \tag{3.20}$$

where $\mu_i$ represents the service rate of mass join operation with $i$ joining participants.

For analytical simplicity, we perform the rest of derivations based on the assumption that Equation 3.20 holds. Then, $P_k$ is expressed as

$$P_k = \left(\frac{\lambda^k}{(\lambda+\mu)^k}\right) P_0 \tag{3.21}$$

Thus, Equation 3.19 is simplified as

$$P_0 = \frac{\mu}{\lambda+\mu} \tag{3.22}$$

Hence, $P_k$ can be generalized as

$$P_k = \frac{\mu\lambda^k}{(\lambda+\mu)^{k+1}} \tag{3.23}$$

We express the average number of batch join size, $Y$, as

$$E[Y] = \sum_{k=0}^{\infty} kP_k \tag{3.24}$$

which is further derived as

$$E[Y] = \sum_{k=0}^{\infty} k\frac{\mu\lambda^k}{(\lambda+\mu)^{k+1}} \tag{3.25}$$

$$E[Y] = \frac{\mu}{\lambda+\mu}\sum_{k=0}^{\infty} k\frac{\lambda^k}{(\lambda+\mu)^k} \tag{3.26}$$

For any real number $0 < x < 1$, the following condition holds [72]:

$$x + 2x^2 + 3x^3 + ... = \frac{x}{(1-x)^2} \tag{3.27}$$

Since $\lambda/(\lambda+\mu) < 1$, we use the Equation 3.27 to simplify Equation 3.26 as

$$E[Y] = \frac{\lambda}{\lambda+\mu}\frac{\frac{\lambda}{\lambda+\mu}}{(1-\frac{\lambda}{\lambda+\mu})^2} \tag{3.28}$$

$$E[Y] = \frac{\lambda^2+\lambda\mu}{\mu^2} \tag{3.29}$$

The average number of joining participants, $E[Y]$, can also be represented by the average number of participants waiting in the queue, $L_q$, because the system joins all the waiting participants at once. By using $L = L_q + E[S]$, we express $L$ as follows:

$$L = \frac{\lambda^2+\lambda\mu}{\mu^2} + \frac{1}{\mu} \tag{3.30}$$

$$L = \frac{\lambda^2+2\lambda\mu}{\mu^2} \tag{3.31}$$

Finally, by applying Little's Law, the average waiting time for the mass join operation is derived as

$$W = \frac{L}{\lambda} \tag{3.32}$$

$$W = \frac{\lambda+2\mu}{\mu^2} \tag{3.33}$$

Note that results we obtain in Equation 3.31 and Equation 3.33 are only valid for a subset of protocols where the number of joining participant has a negligible effect on the group key update time. Generalized results can be obtained by combining the closed-from expression of $P_0$ in Equation 3.19 with the balance equation given in Equation 3.14.

### 3.3. Leave Model

In group key agreement protocols, participants should not be able to take part in the group communication after they leave the group. The group key needs to be updated for preventing the leaving participants to access the encrypted communication. Therefore, many DGKAPs provide a leave operation to update the group key after a participant leave occurs. We now propose Leave Model for the scalability analysis of a leave operation by adding a third system as an output to Residence Model. When a participant requests to leave the group, they are removed from Residence Model and added into Leave Model where they wait until the group key update process is completed. When the group key is updated, the leaving participant is removed from the leave model.

Table 3.3. Definition of the Parameters used in the Leave Model

| **Parameter** | **Explanation** |
|---|---|
| $S_{res}$ | The residence time of participants |
| $S_{leave}$ | The service time of the leave operation |
| $S_{agg}$ | The aggregation of residence time and the service time of the leave operation |
| $\mu = 1/E[S_{res}]$ | The mean residence rate |
| $\mu_{leave} = 1/E[S_{leave}]$ | The mean service rate for the leave operation |
| $\mu_{agg} = 1/E[S_{agg}]$ | The mean aggregated service rate |

In many group key agreement protocols, multiple participants can leave the group simultaneously, without waiting in a queue. The participants do not depend on a single server to leave the group. Instead, each participant depends on a neighbor participant to perform the necessary operations. The likelihood of neighbor participants leaving at the same time is low enough to ignore. Therefore, we assume that participants can always rely on a neighbor participant to perform the leave operation. By using this assumption, we model a leave operation with parallel servers as shown in Figure 3.5.

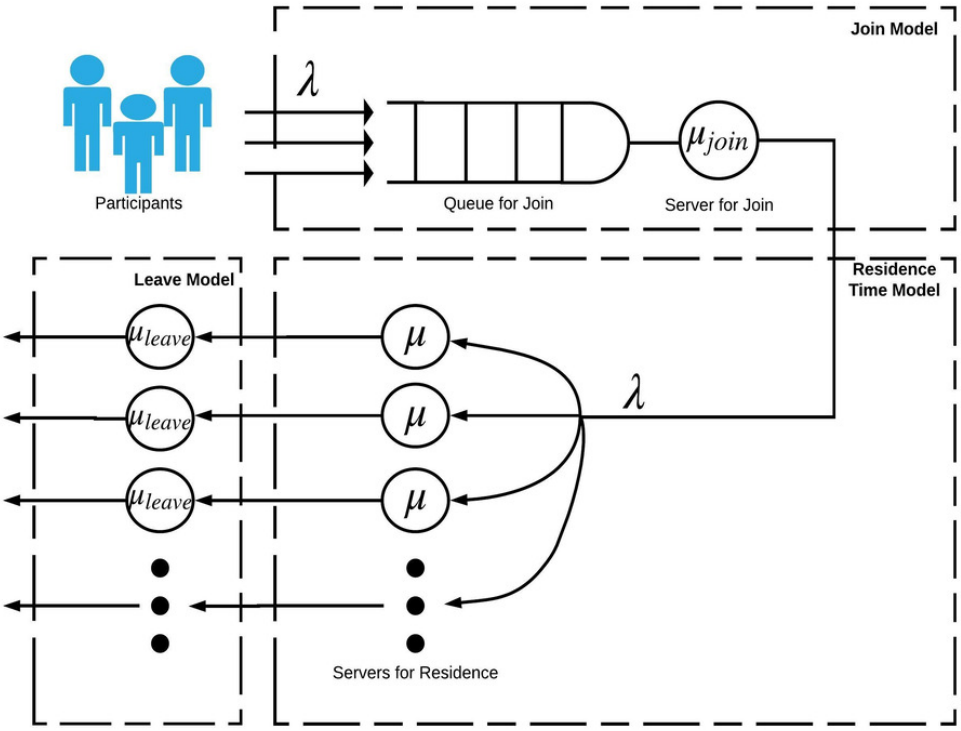



Figure 3.5. Overview of Performance Model with Leave Operation

We aggregate group residence servers and leave servers as a single server with the following service time expression:

$$E[S_{agg}] = E[S_{res}] + E[S_{leave}] \tag{3.34}$$

which implies

$$\frac{1}{\mu_{agg}} = \frac{1}{\mu} + \frac{1}{\mu_{leave}} \tag{3.35}$$

$$\mu_{agg} = \frac{\mu.\mu_{leave}}{\mu + \mu_{leave}} \tag{3.36}$$

where $\mu$ is service rate for residence and $\mu_{leave}$ is the service rate for a leave operation. By using this approach, we remove the leave queue and extend the scope of Residence Model to include a leave operation. Accordingly, the steady-state probability

distributions of Residence Model need to be modified. Equation 3.8 is modified to

$$P_k = \frac{(\frac{\lambda}{\mu_{agg}})^k . e^{-\frac{\lambda}{\mu_{agg}}}}{k!} \tag{3.37}$$

where $\mu_{agg}$ is defined in Equation 3.36. Moreover, we can extend the expected service time of join operation as follows:

$$E[S_{join}] = \frac{E\left[S_{join}^{\left\lceil \frac{\lambda}{\mu_{agg}} \right\rceil}\right] + E\left[S_{join}^{\left\lfloor \frac{\lambda}{\mu_{agg}} \right\rfloor}\right]}{2} \tag{3.38}$$

Updated $E[S_{join}]$ is used for the derivation of performance metrics in Equation 3.12 and 3.13.

## 3.4. Possible Scenarios for Analytical Models

Our analytical models are designed to measure the scalability of secure multi-party group communication systems. We focus on systems where the communication is encrypted by a common key and the set of participants is is dynamic. We propose the following scenario:

**Scenario 3.1** *A private video conference is held by a group of participants. However, each of these participants attends at different time. Once they enter the conference, they want to stay for a while. Eventually, they lose their interest and decide to leave. The conference data is encrypted with a common shared key which is updated each time a participant enters or leaves the conference. Therefore, we make the following assumptions:*

*(i) Updating group key operation causes a small delay for participants.*

*(ii) The time interval between start and end of the conference is much larger than the time interval a participant spends in the conference.*

*(iii) The arrival time of participants is not deterministic.*

*(iv) All of the registered participants are able to attend the conference.*

Our assumption of unknown participant arrival time in Scenario 3.1 allows us to model the arrivals of the join operation as a Poisson process. With Assumption (iv), we can use infinite number of servers to model the number of participants in the conference. Therefore, our performance models are suitable for Scenario 3.1. Another possible scenario is given as follows:

**Scenario 3.2** *Doctors in a hospital establishes a secure multi-party communication channel in order to share confidential data regarding their patients. Doctors communicate with each other via a shared session key which is updated each time a participant joins or leaves the communication. After doctors share information about their patients, they stay in the communication for a while to learn the information provided by other doctors. Then, doctors leave the communication. Therefore, we make the following assumptions:*

*(i) Doctors cannot share information during group key operations.*
*(ii) The number of participants in the communication channel is always greater than two.*
*(iii) The arrival time of doctors is not deterministic.*
*(iv) There is no limit for the number of participants in the channel.*
*(v) Information sharing process takes much more time than updating the group key due to shared medical images such as brain scans.*

Using Assumption (iii) and Assumption (iv), we can model the join operation as an M/M/1 queue and number of doctors in the channel as a M/G/$\infty$ queue, respectively. Therefore, Scenario 3.2 can utilize our performance models.

## 3.5. Attack Models

Scenario 3.1 and Scenario 3.2 presented in Section 3.4 are vulnerable to specific attacks. For instance, in Scenario 3.1, a DDoS attack for the join operation will cause a significant lag for the streamed video. In Scenario 3.2, a similar attack will prevent data sharing among doctors. There are other possible attacks for these scenarios.

We consider three different attack models regarding our performance model. These attack models are called Join Flood Attack Model, Immediate Leaving Attack Model and Disconnector Attack Model.

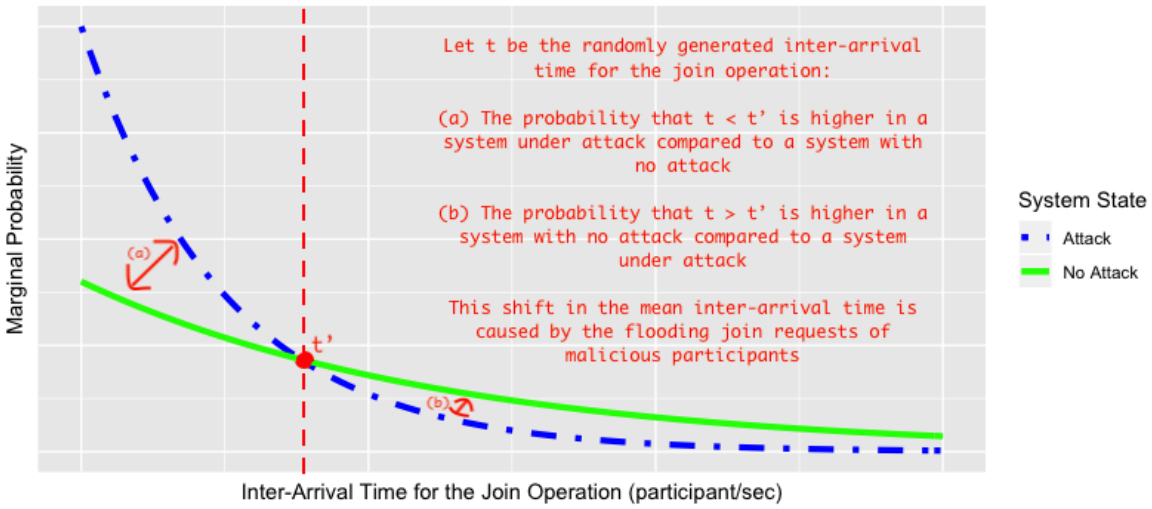



Figure 3.6. An illustration for the change in the inter-arrival time distribution due to the Join Flood Attack Model

The first attack model we propose is the Join Flood Attack Model. In this model, malicious participants are allowed to send join requests to a communication group. However, such participants cannot actually join the group. For instance, malicious participants may request to join the groups in Scenario 3.1 and Scenario 3.2. The frequent join requests may decrease the mean inter-arrival time and cause a shift in the inter-arrival time distribution of the join operation. Figure 3.6 shows an illustration for the change in the inter-arrival time distribution due to the Join Flood Attack Model.

When the frequency of join requests increases, the communication system spends most of the time with group key update operations. As a result, honest participants are not able to communicate with each other. For instance, doctors in Scenario 3.2 cannot share their intelligence when the system is always busy with updating the group key.

As another attacking model, we assume that malicious participants has the ability to join into the communication group, and they leave immediately after they have joined. We name this model as the Immediate Leaving Attack Model. The aim of attackers in Immediate Leaving Attack Model is to keep the server busy with group key update operations. Therefore, this model is considered an extended version of Join Flood Attack Model.

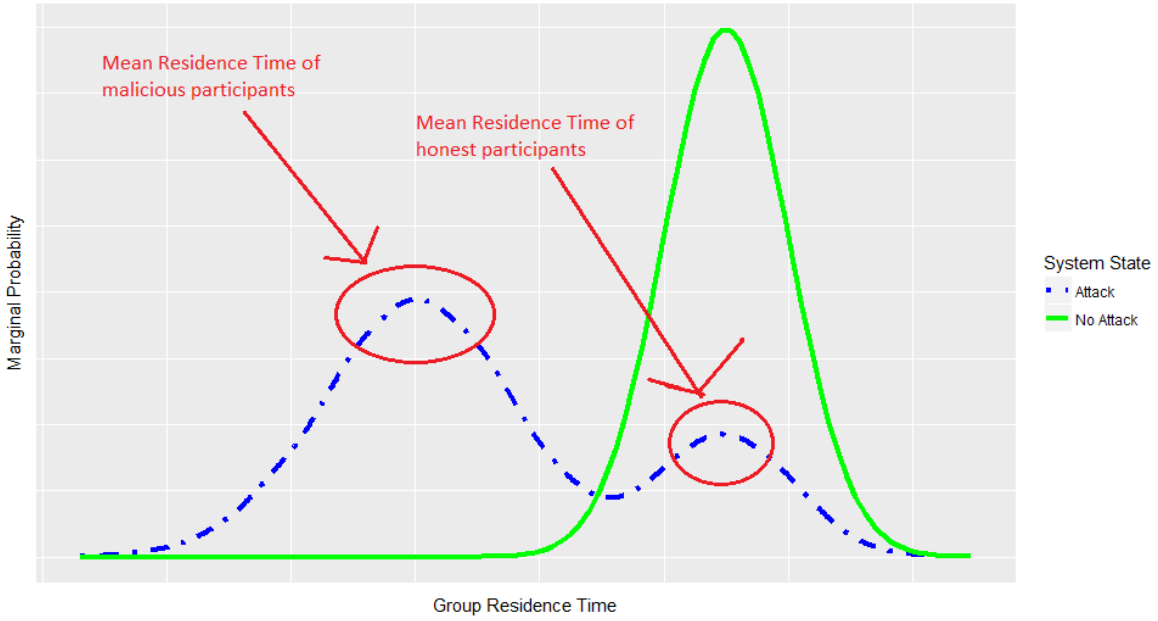



Figure 3.7. An illustration for the change in the residence time distribution due to the Immediate Leaving Attack Model

In the Immediate Leaving Attack Model, various system parameters are affected. First, $\lambda$ increases with join requests of malicious participants. Also, the average waiting time and communication time among participants increases significantly. In addition, $\mu$ increases with the frequency of participant departure. Since the expected residence time of the participants decreases, the number of participants in the system frequently

changes. Such frequent change in the number of participants creates instability and causes the joining queue to fill faster than expected. Figure 3.7 shows an illustration for the change in the residence time distribution due to the Immediate Leaving Attack Model.

The last attack model we design is called the Disconnector Attack Model. The malicious participants cannot join the communication group however; they have the capability of disconnecting honest participants from the system. When honest participants are disconnected unwillingly, they will want to join into the communication again. Therefore, such an attack type increases the rate of join requests. Moreover, the disconnection of participants cause the residence time to drop.

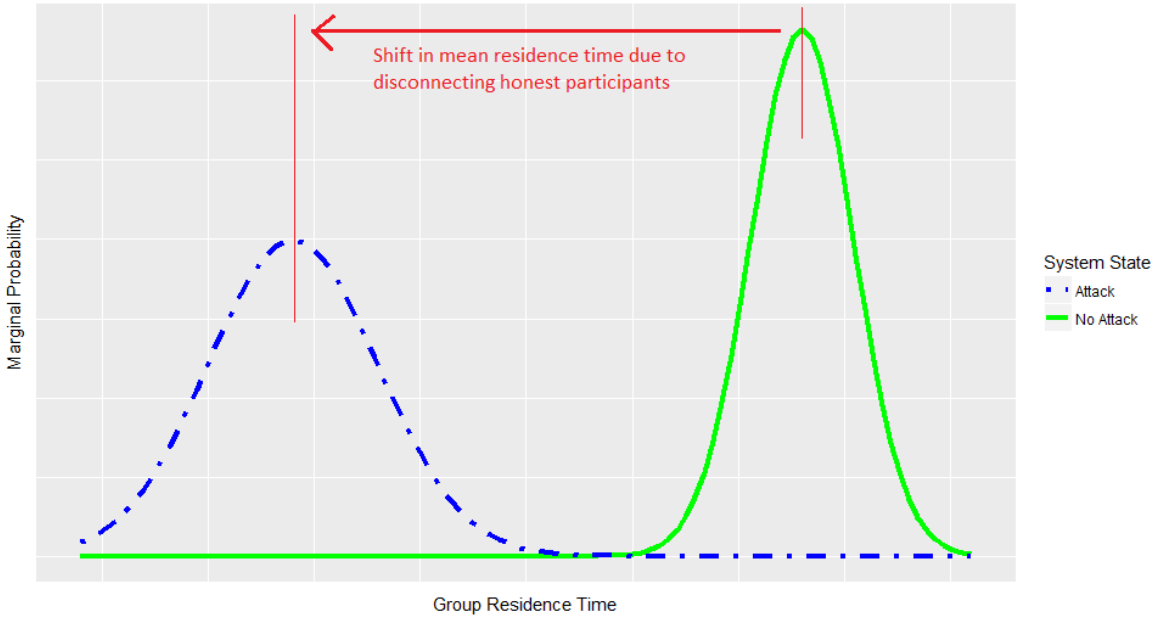



Figure 3.8. An illustration for the change in the residence time distribution due to the Disconnector Attack Model

In the Disconnector Attack Model, $\lambda$ is increased due to the re-connection requests of honest participants. The number of participants change fast in this type of attacks, thus $\mu$ increases. Results are slightly different when compared to Immediate Leaving Attack Model. Figure 3.8 shows an illustration for the change in the residence time distribution due to the Disconnector Attack Model.

We now give a brief summary regarding the effects of the proposed attack models on the participant arrival rate and the residence time. The Join Flood Attack Model may affect the participant arrival rate due to the frequent join requests of malicious participants. Since malicious participants do not join the group, residence time is not affected. The Immediate Leaving Attack Model may affect both the participant arrival rate and the residence time. The participant arrival rate increases due to join requests of malicious participants. In addition, immediate leave of malicious participants decrease the residence time. The Disconnector Attack Model may affect both the participant arrival rate and the residence time. The participant arrival rate is increased due to the re-connection requests of honest participants. Moreover, the residence time is decreased due to the disconnecting honest participants. In overall, all three attack models represent possible attacks that endanger the availability of the communications in the proposed scenarios. The attack models may disrupt the communications by affecting the system parameters. A formal performance model can be used to measure the availability boundaries of a system and provide counter measures to such attack models.

# 4. ANALYSIS OF PROPOSED ANALYTICAL MODELS ON AN EXAMPLE GROUP KEY AGREEMENT PROTOCOL

In this chapter, we present a use case of our performance model on Key Agreement Protocol with Partial Backward Confidentiality (KAP-PBC)[11]. We analyze the performance of join, mass join and join-PBC operations in KAP-PBC. For operations in KAP-PBC, four main functions are used:

- *send-public-key*: each participant computes the temporary public key and broadcasts it.
- *send-secret-key*: each participant computes the secret session key and broadcasts it.
- *verify-public-key*: each participant verifies the temporary public keys of other participants.
- *verify-secret-key*: each participant verifies the the shared secret keys of other participants.

Table 4.1. KAP-PBC functions and their notations

| **Function** | **Parameter Symbol** |
|---|---|
| Temporary Public Key Computation | $t_{pkc}$ |
| Temporary Public Key Verification | $t_{pkv}$ |
| Temporary Public Key Broadcast | $t_{pkb}$ |
| Session Key Computation | $t_{skc}$ |
| Session Key Verification | $t_{skv}$ |
| Session Key Broadcast | $t_{skb}$ |
| Group Key Computation | $t_{gkc}$ |

Let $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ be the set of participants in the group and $P_{n+1}, \ldots, P_{n+m}$ be the joining participants. Then, the following functions are executed while adding new participants into the group:

- Participants $P_n, P_{n+1}, ..., P_{n+m}$ execute send-public-key function.
- Participants $P_{n-1}, P_n, P_{n+1}, ..., P_{n+m}$ execute send-secret-key function.
- All participants, $P_1, ..., P_{n+m}$ execute verify-public-key and verify-secret-key functions.

## 4.1. Analysis of the Single Join Operation

The expression of service time in terms of the notation given in Table 4.1 can be expressed as:

$$\begin{aligned} E[S^{n,m}] &= t_{pkv}(m+1) + t_{skv}(m+2) + t_{skc} + t_{skb}(n+m-1) + t_{pkc} + \\ & t_{pkb}(n+m-1) + t_{gkc} \end{aligned} \tag{4.1}$$

Since $m = 1$ in a single join operation, the expression is simplified as

$$E[S^n] = 2t_{pkv} + 3t_{skv} + t_{skc} + nt_{skb} + t_{pkc} + nt_{pkb} + t_{gkc} \tag{4.2}$$

Performance metrics of the join operation are derived in Equation 3.12 and 3.13 as follows:

$$W = \frac{1}{\frac{1}{E[S_{join}]} - \lambda} \tag{4.3}$$

$$L = \frac{\lambda}{\frac{1}{E[S_{join}]} - \lambda} \tag{4.4}$$

where the average service time of a single join operation, $E[S_{join}]$, is derived as The expected service time of a single join operation with $n$ existing participants is equal to the following equation:

$$E[S^n] = 2t_{pkv} + 3t_{skv} + t_{skc} + nt_{skb} + t_{pkc} + nt_{pkb} + t_{gkc} \tag{4.5}$$

The expected service time of a single join operation is expressed as

$$E[S_{join}] = \frac{E\left[S_{join}^{\left\lceil \frac{\lambda}{\mu} \right\rceil}\right] + E\left[S_{join}^{\left\lfloor \frac{\lambda}{\mu} \right\rfloor}\right]}{2} \tag{4.6}$$

where $\lambda$ is the arrival rate of joining participants and $\mu$ is the service rate of Residence Model. We combine these two equations to obtain a closed-form expression of the average service time for the join operation as follows:

$$E[S_{join}] = \frac{4t_{pkv} + 6t_{skv} + 2t_{skc} + \left\lceil \frac{\lambda}{\mu} \right\rceil t_{skb} + 2t_{pkc} + \left\lceil \frac{\lambda}{\mu} \right\rceil t_{pkb} + 2t_{gkc} + \left\lfloor \frac{\lambda}{\mu} \right\rfloor t_{skb} + \left\lfloor \frac{\lambda}{\mu} \right\rfloor t_{pkb}}{2} \tag{4.7}$$

$$E[S_{join}] = 2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} + (t_{skb} + t_{pkb})\frac{\left\lfloor \frac{\lambda}{\mu} \right\rfloor + \left\lceil \frac{\lambda}{\mu} \right\rceil}{2} \tag{4.8}$$

$$E[S_{join}] = 2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} + (t_{skb} + t_{pkb})\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right) \tag{4.9}$$

We combine Equation 4.3 and 4.1 to derive the average waiting time of a single join operation as

$$W = \frac{2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} + (t_{skb} + t_{pkb})\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)}{1 - \lambda(2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} + (t_{skb} + t_{pkb})\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)} \tag{4.10}$$

Then, by using Little's Law, the average number of participants waiting to join the group is given by

$$L = \frac{\lambda\left(2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} + (t_{skb} + t_{pkb})\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)\right)}{1 - \lambda(2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} + (t_{skb} + t_{pkb})\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)} \tag{4.11}$$

## 4.2. Analysis of the Mass Join Operation

In the mass join operation, the service time depends on both the number of existing participants and the number of joining participants. Therefore, the expression for the mean service time is expressed as

$$\begin{aligned} E[S] =& t_{pkv}(E[m]+1) + t_{skv}(E[m]+2) + t_{skc} + \\ & t_{skb}(E[n]+E[m]-1) + t_{pkc} + t_{pkb}(E[n]+E[m]-1) + t_{gkc} \end{aligned} \tag{4.12}$$

The expected number of existing participants, $E[n]$, is equal to $\lambda/\mu$. In the mass join operation, the expected number of joining participants, $E[m]$, is equal to the number of participants waiting in the queue, denoted as $L_q$. Thus, the following equation holds:

$$\begin{aligned} E[S] =& t_{pkv}(E[m]+1) + t_{skv}(E[m]+2) + t_{skc} + \\ & t_{skb}(E[n]+E[m]-1) + t_{pkc} + t_{pkb}(E[n]+E[m]-1) + t_{gkc} \end{aligned} \tag{4.13}$$

$$\begin{aligned} E[S] =& t_{pkv}(L_q+1) + t_{skv}(L_q+2) + t_{skc} + \\ & t_{skb}\left(\frac{\lambda}{\mu}+L_q-1\right) + t_{pkc} + t_{pkb}\left(\frac{\lambda}{\mu}+L_q-1\right) + t_{gkc} \end{aligned} \tag{4.14}$$

$$\begin{aligned} E[S] =& \frac{\lambda}{\mu}(t_{skb}+t_{pkb}) + L_q(t_{pkv}+t_{pkb}+t_{skb}+t_{skv}) + \\ & t_{pkv} + 2t_{skv} + t_{skc} + t_{pkc} + t_{gkc} - t_{skb} - t_{pkb} \end{aligned} \tag{4.15}$$

For simplicity, we introduce a new variable $\tau_1$ which is given by

$$\tau_1 = t_{pkv} + 2t_{skv} + t_{skc} + t_{pkc} + t_{gkc} - t_{skb} - t_{pkb} \tag{4.16}$$

Then, Equation 4.15 is simplified as

$$E[S] = \frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 \tag{4.17}$$

where $\tau_1$ is given by

$$\tau_1 = t_{pkv} + 2t_{skv} + t_{skc} + t_{pkc} + t_{gkc} - t_{skb} - t_{pkb} \tag{4.18}$$

By the definition of mass join operation, a participant only waits in the queue until the current mass join operation is completed. If a participant arrives when the mass join operation is started, the time spent in the queue will be equal to the service time of the queue. On the other hand, if a participant arrives, at time $\tau$, time after a mass join operation is started, the waiting time in the queue will be less than the service time. Therefore, the following inequality holds for all conditions:

$$W_q \leq E[S] \tag{4.19}$$

By combining inequality in Equation 4.19 and the expression in Equation 4.17, we derive an upper bound for $W_q$ as follows:

$$W_q \leq \frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 \tag{4.20}$$

$$W_q \leq \frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \lambda W_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 \tag{4.21}$$

$$W_q(1 - \lambda(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})) \leq \frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \tau_1 \tag{4.22}$$

$$W_q \leq \frac{\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \tau_1}{1 - \lambda(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{4.23}$$

$$W_q \leq \frac{\lambda(t_{skb} + t_{pkb}) + \mu\tau_1}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{4.24}$$

where $\lambda$ is the participant arrival rate for the mass join operation and $\mu$ is the service rate of Residence Model. By using Little's Law, an upper bound for the number of

participants in the queue is derived as

$$L_q \leq \frac{\lambda^2(t_{skb} + t_{pkb}) + \lambda\mu\tau_1}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{4.25}$$

## 4.3. Analysis of the join-PBC Operation

KAP-PBC supports both join and mass join operations for allowing new participants to join the group. Moreover, KAP-PBC also provides a third and novel approach for joining new participants, namely join-PBC. In join-PBC, a chosen joining participant computes the group key just before the join operation. This process is considered as a single join operation. Then, joining participants are added into the group with a mass join operation and then a new key is computed. Therefore, we consider the join-PBC operation as a pipeline process of a single join and a mass join operation as shown in Fig 4.1.

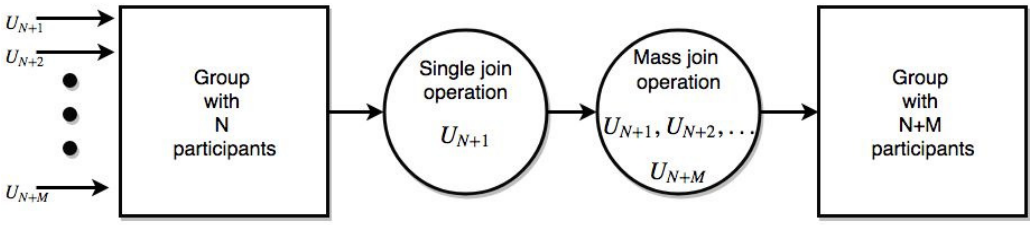



Figure 4.1. An overview of join-PBC operation in KAP-PBC

join-PBC operation is useful when KAP-PBC is integrated into a secure file sharing system that stores files in a confidential manner. The chosen participant downloads and decrypts the stored files with the previous key just before joining the group. Then, the same participant encrypts these files with the new key and uploads them back into the storage system.

The analysis of the join-PBC operation can be accomplished by using the analysis of the mass join and the single join operations. The mean service time is expressed as

$$E[S] = \frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) +$$
$$t_{pkv} + 2t_{skv} + t_{skc} + t_{pkc} + t_{gkc} - t_{skb} - t_{pkb} + E[S_{join}] \quad (4.26)$$
$$(4.27)$$

Then, it is derived as

$$E[S] = \frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) +$$
$$t_{pkv} + 2t_{skv} + t_{skc} + t_{pkc} + t_{gkc} - t_{skb} - t_{pkb} +$$
$$2t_{pkv} + 3t_{skv} + t_{skc} + \frac{\lambda}{\mu}t_{skb} + t_{pkc} + \frac{\lambda}{\mu}t_{pkb} + t_{gkc} \quad (4.28)$$
$$E[S] = 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 +$$
$$2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} \quad (4.29)$$

For a simpler notation, we introduce a variable $\tau_2$ which is given by

$$\tau_2 = 2t_{pkv} + 3t_{skv} + t_{skc} + t_{pkc} + t_{gkc} \quad (4.30)$$

Then, Equation 4.29 is simplified as

$$E[S] = 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 + \tau_2 \quad (4.31)$$

By substituting the expression in Equation 4.31 into Equation 4.19, we derive an upper bound for $W_q$ in the join-PBC operation

$$W_q \leq \frac{2\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 + \tau_2 \tag{4.32}$$

$$W_q \leq 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \lambda W_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 + \tau_2 \tag{4.33}$$

$$W_q(1 - \lambda(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})) \leq 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \tau_1 + \tau_2 \tag{4.34}$$

$$W_q \leq \frac{2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \tau_1 + \tau_2}{1 - \lambda(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{4.35}$$

$$W_q \leq \frac{2\lambda(t_{skb} + t_{pkb}) + \mu(\tau_1 + \tau_2)}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{4.36}$$

Then, by applying Little's Law, an upper bound for $L_q$ in the join-PBC operation is derived as

$$L_q \leq \frac{2\lambda^2(t_{skb} + t_{pkb}) + \lambda\mu(\tau_1 + \tau_2)}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{4.37}$$

## 4.4. Analysis of the Leave Operation

In this section, we perform an analysis of the leave operation in KAP-PBC. We extend the definition of the residence time such that the residence time covers the leave time of a participant. Then, the expected value of the extended residence time is given by

$$E[S_{agg}] = E[S_{res}] + E[S_{leave}] \tag{4.38}$$

as expressed in Equation 3.36, $E[S_{res}]$ depends on the application type and environment. Therefore, we assume that the value of $E[S_{res}]$ is constant for the analysis. $E[S_{leave}]$ depends on the number of existing participants and computational power of these participants. We focus on $E[S_{leave}]$ for the rest of the analysis.

When there are $n$ existing participants in the group, $P_1, ..., P_n$, the functions executed by participants during the leave of $P_i$ are as follows:

- Participant $P_{i-1}$ executes send-public-key function.
- Participants $P_{i-1}$ and $P_{i-2}$ execute send-secret-key functions.
- All participants except $P_i$ executes verify-public-key and verify-secret-key functions.

Computations are performed individually by each participant. We consider the participant with the most time consuming operations as the service time of a leave operation. Participant $P_{i-1}$ needs to perform more computations based on function executions.

$$E[S_{leave}] = t_{pkv} + t_{skv} + t_{pkc} + t_{skc} + t_{pkb}(n-2) + t_{skb}(n-2) + t_{gkc} \tag{4.39}$$

Since $E[n] = \frac{\lambda}{\mu_{agg}}$, from Equation 3.9, we express $E[S_{leave}]$ as

$$E[S_{leave}] = t_{pkv} + t_{skv} + t_{pkc} + t_{skc} + t_{pkb}\left(\frac{\lambda}{\mu_{agg}} - 2\right) + t_{skb}\left(\frac{\lambda}{\mu_{agg}} - 2\right) + t_{gkc} \tag{4.40}$$

The expressions $E[S_{leave}]$ and $\mu_{agg}$ are related by the following equations:

$$E[S_{agg}] = E[S_{res}] + E[S_{leave}] \tag{4.41}$$

$$\frac{1}{\mu_{agg}} = \frac{1}{\mu} + \frac{1}{\mu_{leave}} \tag{4.42}$$

$$\mu_{agg} = \frac{1}{\frac{1}{\mu} + \frac{1}{\mu_{leave}}} \tag{4.43}$$

$$\mu_{agg} = \frac{\mu.\mu_{leave}}{\mu + \mu_{leave}} \tag{4.44}$$

We substitute the value of $\mu_{agg}$ into Equation 4.40 to obtain

$$E[S_{leave}] = t_{pkv} + t_{skv} + t_{pkc} + t_{skc} + t_{pkb}\left(\frac{\lambda}{\frac{\mu.\mu_{leave}}{\mu+\mu_{leave}}} - 2\right) + t_{skb}\left(\frac{\lambda}{\frac{\mu.\mu_{leave}}{\mu+\mu_{leave}}} - 2\right) + t_{gkc} \tag{4.45}$$

$$\begin{aligned} E[S_{leave}] &= t_{pkv} + t_{skv} + t_{pkc} + t_{skc} + \\ &+ t_{pkb}\left(\frac{\lambda\left(\mu + \mu_{leave}\right)}{\mu.\mu_{leave}} - 2\right) + t_{skb}\left(\frac{\lambda(\mu + \mu_{leave})}{\mu.\mu_{leave}} - 2\right) + t_{gkc} \end{aligned} \tag{4.46}$$

$$\begin{aligned} \frac{1}{\mu_{leave}} &= t_{pkv} + t_{skv} + t_{pkc} + t_{skc} + \\ &+ t_{pkb}\left(\frac{\lambda\left(\mu + \mu_{leave}\right)}{\mu.\mu_{leave}} - 2\right) + t_{skb}\left(\frac{\lambda\left(\mu + \mu_{leave}\right)}{\mu.\mu_{leave}} - 2\right) + t_{gkc} \end{aligned} \tag{4.47}$$

$$\begin{aligned} \frac{1}{\mu_{leave}} &= t_{pkv} + t_{skv} + t_{pkc} + t_{skc} - 2t_{pkb} - 2t_{skb} + t_{gkc} \\ &+ \frac{\lambda\left(\mu + \mu_{leave}\right)t_{pkb}}{\mu.\mu_{leave}} + \frac{\lambda\left(\mu + \mu_{leave}\right)t_{skb}}{\mu.\mu_{leave}} \end{aligned} \tag{4.48}$$

For simplicity, we denote the summation of constant terms as

$$\tau_3 = t_{pkv} + t_{skv} + t_{pkc} + t_{skc} - 2t_{pkb} - 2t_{skb} + t_{gkc} \tag{4.49}$$

Then, Equation 4.48 is simplified as follows:

$$\frac{1}{\mu_{leave}} = \frac{\lambda\left(\mu + \mu_{leave}\right)t_{pkb}}{\mu.\mu_{leave}} + \frac{\lambda\left(\mu + \mu_{leave}\right)t_{skb}}{\mu.\mu_{leave}} + \tau_3 \tag{4.50}$$

$$\frac{1 - \tau_3.\mu_{leave}}{\mu_{leave}} = \frac{\lambda\left(\mu + \mu_{leave}\right)t_{pkb}}{\mu.\mu_{leave}} + \frac{\lambda\left(\mu + \mu_{leave}\right)t_{skb}}{\mu.\mu_{leave}} \tag{4.51}$$

$$1 - \tau_3\mu_{leave} = \frac{\lambda\left(\mu + \mu_{leave}\right)t_{pkb}}{\mu} + \frac{\lambda\left(\mu + \mu_{leave}\right)t_{skb}}{\mu} \tag{4.52}$$

$$\mu - \mu.\tau_3.\mu_{leave} = \lambda.\mu.(t_{pkb} + t_{skb}) + \lambda.\mu_{leave}(t_{pkb} + t_{skb}) \tag{4.53}$$

$$\mu.(1 - \lambda(t_{pkb} + t_{skb})) = \mu_{leave}.(\mu\tau_3 + t_{pkb} + t_{skb}) \tag{4.54}$$

$$\mu_{leave} = \frac{\mu.(1 - \lambda(t_{pkb} + t_{skb}))}{\mu.\tau_3 + t_{pkb} + t_{skb}} \tag{4.55}$$

Similarly, $E[S_{leave}]$ is calculated as $1/\mu_{leave}$, which reduces to

$$E[S_{leave}] = \frac{\mu.\tau_3 + t_{pkb} + t_{skb}}{\mu.(1 - \lambda(t_{pkb} + t_{skb}))} \tag{4.56}$$

## 4.5. Numerical Results for the Example KAP-PBC Use Case Scenario

In this section, we obtain the numerical results by using our derived expressions with the execution time we measured for KAP-PBC functions. Results are obtained via simulations which are carried out in a computer with a 1.8 GHz Intel Core i5 processor and 4GB RAM. Execution times of KAP-PBC operations are reported in Table 4.2.

Table 4.2. KAP-PBC functions and their execution times on a computer with 1.8 GHz Intel Core i5 processor, for AES-256 (CBC mode) as the cipher

| **Function** | **Execution Time (msec)** | **Parameter Symbol** |
|---|---|---|
| Temporary Public Key Computation | 536 | $t_{pkc}$ |
| Temporary Public Key Verification | 6 | $t_{pkv}$ |
| Temporary Public Key Broadcast | 1 | $t_{pkb}$ |
| Session Key Computation | 12 | $t_{skc}$ |
| Session Key Verification | 20 | $t_{skv}$ |
| Session Key Broadcast | 2 | $t_{skb}$ |
| Group Key Computation | 10 | $t_{gkc}$ |

For the join operation, derived performance metrics are expressed in Equation 4.10 and 4.11. We substitute values in Table 4.2 into these equations and derive numerical results as follows:

$$W = \frac{2 * 6 + 3 * 20 + 12 + 536 + 10 + (2 + 1)\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)}{1 - \lambda(2 * 6 + 3 * 20 + 12 + 536 + 10 + (2 + 1)\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)} \tag{4.57}$$

$$W = \frac{631.5 + 3\left\lfloor \frac{\lambda}{\mu} \right\rfloor}{1 - \lambda\left(631.5 + 3\left\lfloor \frac{\lambda}{\mu} \right\rfloor\right)} \tag{4.58}$$

$$L = \frac{\lambda\left(2 * 6 + 3 * 20 + 12 + 536 + 10 + (2 + 1)\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)\right)}{1 - \lambda(2 * 6 + 3 * 20 + 12 + 536 + 10 + (2 + 1)\left(\left\lfloor \frac{\lambda}{\mu} \right\rfloor + 0.5\right)} \tag{4.59}$$

$$L = \frac{631.5\lambda + 3\lambda\left\lfloor \frac{\lambda}{\mu} \right\rfloor}{1 - \lambda\left(631.5 + 3\left\lfloor \frac{\lambda}{\mu} \right\rfloor\right)} \tag{4.60}$$

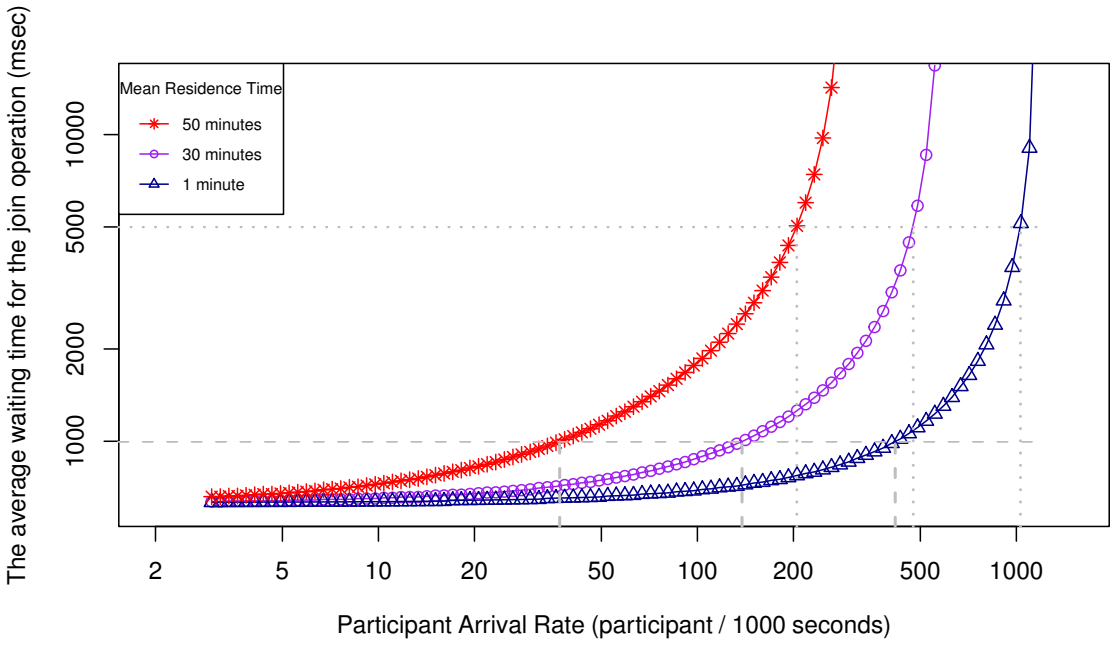



Figure 4.2. Graph of Average Waiting Time in KAP-PBC against Participant Arrival Rate for different Mean Residence Time

The average waiting time in the join operation with respect to the participant arrival rate is shown in Figure 4.2. The figure displays three curves with different mean residence times: i) 1 minute, ii) 10 minutes and iii) 50 minutes. During the numerical analysis, we assume that KAP-PBC is used to provide security in an instant messaging application where participants are able to join a group in a few seconds. Therefore, we assign 1000 milliseconds and 5000 milliseconds as threshold values for the average waiting time. Then, for each threshold value, we try to find the maximum participant arrival rate that does not violate the threshold. For the threshold of 1000 milliseconds, the limiting participant arrival rates are 37 participants, 138 participants and 417 participants at every 1000 seconds for the corresponding cases with 1 minute, 10 minutes and 50 minutes mean residence time, respectively. For the threshold of 5000 milliseconds, the limiting participant rates are 205 participants, 475 participants and 1030 participants at every 1000 seconds for the corresponding cases with 1 minute, 10 minutes and 50 minutes mean residence time, respectively. We infer that an application with more than 500 participant arrivals at every 1000 seconds, causes significantly long average waiting time for the joining participants. Moreover, we observe that the mean residence time of participants have a significant effect on the average waiting time. When the mean residence time is around a minute, the average waiting time exceeds 1000 milliseconds with more than 500 participant arrivals at every 1000 seconds, which is a high value for a messaging application. Therefore, we conclude that KAP-PBC can be used in applications with short mean residence time or with infrequent participant arrival rates. In such applications, the average waiting time is expected to be short. On the other hand, using KAP-PBC in applications with frequent participant arrival rates and long mean residence time may yield long average waiting time for the joining participants.

For the mass join operation, upper bounds for performance metrics are expressed in Equation 4.24 and 4.25. By substituting the values in Table 4.2 into these equations, we obtain

$$W_q \leq \frac{\lambda(2+1) + \mu(6 + 2 * 20 + 12 + 536 + 10 - 2 - 1)}{\mu - \lambda\mu(6 + 1 + 2 + 20)} \tag{4.61}$$

$$W_q \leq \frac{3\lambda + 601\mu}{\mu - 29\lambda\mu} \tag{4.62}$$

$$L_q \leq \frac{\lambda^2(2+1) + \lambda\mu(6 + 2 * 20 + 12 + 536 + 10 - 2 - 1)}{\mu - \lambda\mu(6 + 1 + 2 + 20)} \tag{4.63}$$

$$L_q \leq \frac{3\lambda^2 + 601\lambda\mu}{\mu - 29\lambda\mu} \tag{4.64}$$

For the join-PBC operation, upper bounds for performance metrics are expressed in Equation 4.36 and 4.37. We substitute values in Table 4.2 into these equations to obtain

$$W_q \leq \frac{2\lambda(2+1) + \mu(601 + 630)}{\mu - \lambda\mu(6 + 1 + 2 + 20)} \tag{4.65}$$

$$W_q \leq \frac{6\lambda + 1231\mu}{\mu - 29\lambda\mu} \tag{4.66}$$

$$L_q \leq \frac{2\lambda^2(2+1) + \lambda\mu(601 + 630)}{\mu - \lambda\mu(6 + 1 + 2 + 20)} \tag{4.67}$$

$$L_q \leq \frac{6\lambda^2 + 1231\lambda\mu}{\mu - 29\lambda\mu} \tag{4.68}$$

# 5. ANALYTICAL MODELS FOR THE SCALABILITY ANALYSIS OF SECURE FILE SHARING SYSTEMS WITH A DEMONSTRATIVE USE CASE SCENARIO

In this section, we extend our performance model to be used for secure file sharing systems (SFSS). First, we list our assumptions for the extension process. Then, we provide a demonstrative use case scenario by employing Private File Sharing System (PFSS)[11] as an example secure file sharing system. We reduce the number of possible scenarios by using the following assumptions:

- **Assumption 5.1** Encryption/Decryption Time Complexity. *The time required to encrypt or decrypt a file depends only on the size of the file.*
- **Assumption 5.2** Network Environment of Participants. *Participants in a group share the same characteristics in terms of networking capability. Participants have more or less the same Internet connection bandwidth.*
- **Assumption 5.3** Constant Internet Connection Bandwidth. *The change in Internet connection bandwidth of participants is relatively small compared to the computational time complexity of the group key update operation.*

## 5.1. Performance Model for Secure File Sharing System Operations by Using PFSS as a Demonstrative Use Case Scenario

PFSS is a secure file sharing system that provides file encryption, decryption, upload and download operations. PFSS utilizes Key Agreement Protocol with Partial Backward Confidentiality (KAP-PBC). During the performance modelling of PFSS, we consider the service time as the combination of the file transmission time and the file encryption time.

In PFSS, files are stored in an encrypted manner. AES-256 in CBC mode is used to provide confidentiality. Assumption 1 holds because AES has linear time complexity

with respect to the size of encrypted files. In addition, with Assumption 2, the time required for a file transmission can be expressed for each participant as follows:

$$t_F = \frac{|\mathcal{F}|}{\mathcal{B}} \tag{5.1}$$

where $|\mathcal{F}|$ is the size of the transmitted file and $\mathcal{B}$ is the network connection bandwidth. File transmission time, $t_F$, depends on the Internet connection bandwidth and the size of the transmitted file. Due to Assumption 3, $\mathcal{B}$ is considered as a constant value and $t_F$ is linearly proportional to the size of the file. The transmission time and the file size have the same underlying probability distribution. Therefore, service time can be modelled by a file size distribution.

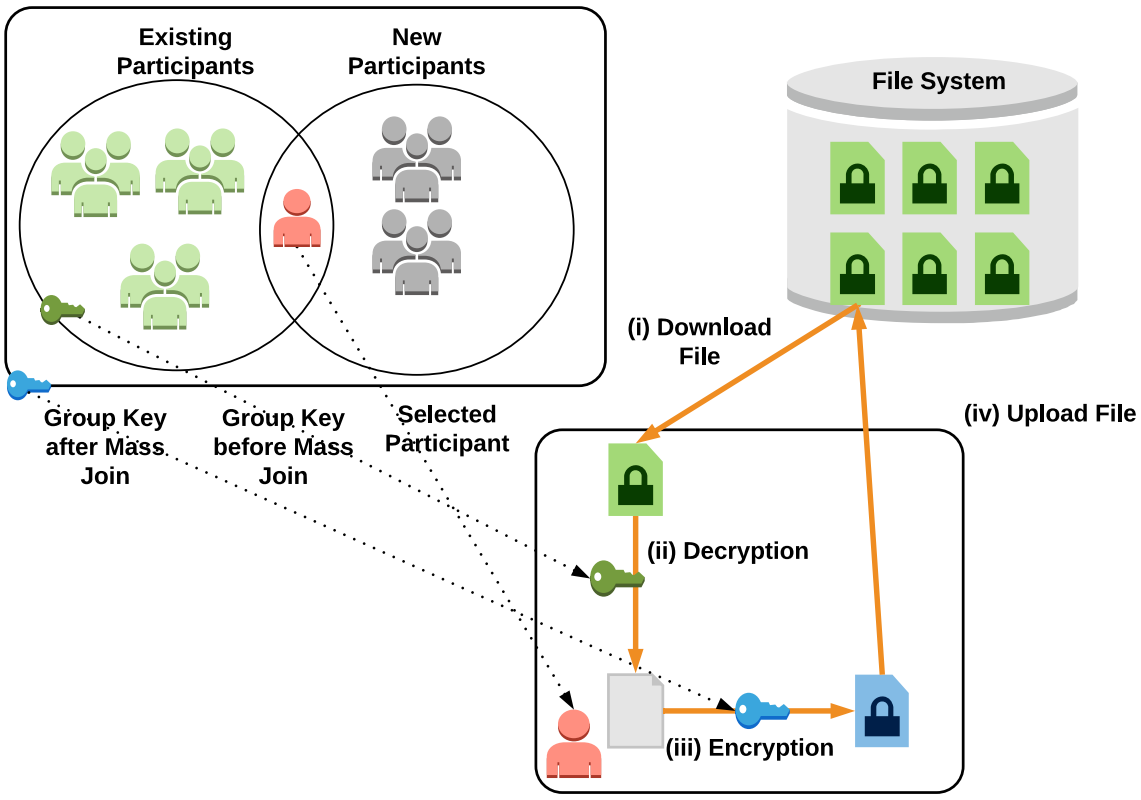



Figure 5.1. An Illustration of File Re-Encryption Process in PFSS

PFSS uses join-PBC function of KAP-PBC protocol to add new participants into the group. While executing the function, one of the new participants is selected for computing the group key just before joining the group. Therefore, this participant is able to re-encrypt shared group file(s) as shown in Figure 5.1. First, the selected

participant joins the group by computing the group key just before joining the group. Then, all participants compute the new group key. Later, the selected participant re-encrypts the shared files by following steps shown in the Figure 5.1 The process is displayed in four steps: (i) downloads the shared group file(s), (ii) decrypts the file by using the existing group key, (iii) encrypts the file by using the new group key, (iv) uploads the re-encrypted file into the file system of PFSS.

In order to make more realistic numerical evaluations, we also consider studies in the literature regarding the distribution of file size such as [73, 74, 75, 76]. In general, the distribution of the size of a file can be expressed by using Pareto and Lognormal distributions. Therefore, we model the service time based on these distributions. Since Pareto and Lognormal distributions are non-Markovian, we assume that the service time is generally distributed. For simplicity, we also assume that the inter-arrival time is exponentially distributed. Therefore, we can use an $M/G/1$ queue for the file sharing operations as a general case. To deliver a more concrete example, we come up with a scenario and obtain numerical performance results based on this scenario as given in the following subsection.

## 5.2. A Use-Case Scenario for Healthcare System

We present an example use-case scenario for the use of Private File Sharing System [11] as a Healthcare System that securely stores medical records of patients. In this scenario, we assume that medical records of each patient are stored as an encrypted file in PFSS and each file is associated with a group of medical personnel. We also assume that an average size of files is approximately 300 MB as reported in [77]. Therefore, the chosen joining participant in our scenario re-encrypts a 300 MB file after new participants join the group.

Performance metrics of PFSS in this scenario depend significantly on the Internet connection bandwidth. Consequently, we categorize the network connection bandwidth capacity into three different ranges for the analysis:

(i) *Less than 10 Mbps:* In this case, the file download and the file upload operations are the bottleneck due to negligible group key update time. Since both the file size and the bandwidth are assumed as constant values, we use an $M/D/1$ queue to model upload and download operations. The expected service time is calculated as the time required to upload and download a medical file size, given by

$$E[S] = \frac{2\ x\ 300x10^6}{\mathcal{B}} = \frac{6x10^8}{\mathcal{B}}\ (sec)\ = \frac{6x10^{11}}{\mathcal{B}}\ (msec) \tag{5.2}$$

where $\mathcal{B}$ is the Internet connection bandwidth in terms of bits per second (bps). As a result, the service rate is expressed as:

$$\mu = \frac{1}{E[S]} = \frac{\mathcal{B}}{6x10^{11}} \tag{5.3}$$

Note that performance metrics of an $M/G/1$ queue is derived in Pollaczek-Khinchine formula:

$$L\ = \frac{\rho^2 + \lambda^2\sigma_s^2}{2(1-\rho)} + \rho \tag{5.4}$$

$$W\ = \frac{\rho + \lambda\mu\sigma_s^2}{2(\mu-\lambda)} + \frac{1}{\mu} \tag{5.5}$$

where $\rho$ equals to $\lambda/\mu$, $\lambda$ is the arrival rate, $\mu$ is the service rate and $\sigma_s^2$ is the variance of service time. By using the Pollaczek-Khinchine Formula with $\sigma_s^2 = 0$, we derive the mean number of waiting participants of $M/D/1$ queue as

$$L = \frac{\lambda^2}{2\mu(\mu-\lambda)} + \frac{\lambda}{\mu} \tag{5.6}$$

and the mean waiting time as

$$W = \frac{\lambda}{2\mu(\mu-\lambda)} + \frac{1}{\mu} \tag{5.7}$$

(ii) *Between 10 Mbps and 1 Gbps:* File operations and key update mechanism require approximately the same amount of time. File operations are modelled by an

$M/D/1$ queue whereas the key update process, joinPBC operation, is modelled by an $M/M/1$ queue. The aggregation of one deterministic and one exponential service time is closer to exponential distribution. Therefore, we use an $M/M/1$ queue to obtain a lower bound for performance metrics. The service time of the join operation is derived in Equation 4.29. We add the expected time required for the file download/upload and the file encryption/decryption operations into the expression in Equation 4.29 to obtain

$$E[S] = 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 + \tau_2 + t_{upl+dow} + t_{enc+dec} \tag{5.8}$$

where $t_{upl+dow}$ represents the total amount of time required to upload and download a file, and $t_{enc+dec}$ represents the total amount of time required to encrypt and decrypt a file. For the average file size used in healthcare scenario, $t_{upl+dow}$ is

$$t_{upl+dow} = \frac{6x10^{11}}{\mathcal{B}}\ (msec) \tag{5.9}$$

where $\mathcal{B}$ is the network connection bandwidth. Then, the mean service time is expressed by

$$E[S] = 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 + \tau_2 + \frac{6x10^{11}}{\mathcal{B}} + t_{enc+dec} \tag{5.10}$$

Later, by combining the inequality in Equation 4.36 and Equation 5.10, we derive an upper bound for the average waiting time of joining participants as given below:

$$W_q \leq \frac{2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + \tau_1 + \tau_2 + \frac{6x10^{11}}{\mathcal{B}} + t_{enc+dec}}{1 - \lambda(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{5.11}$$

$$W_q \leq \frac{2\lambda(t_{skb} + t_{pkb}) + \mu(\tau_1 + \tau_2 + \frac{6x10^{11}}{\mathcal{B}} + t_{enc+dec})}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{5.12}$$

where $\mu$ is the service rate of Residence Model, $\lambda$ is the arrival rate of joining participants, and $\mathcal{B}$ is the Internet connection bandwidth. Also, we use $\tau 1$ and $\tau 2$ from Equation 4.18 and 4.30, respectively. Then, by using $L_q = W_q\lambda$, we derive an upper bound for the average number of joining participants waiting in the queue:

$$W_q\lambda \leq \lambda\frac{2\lambda(t_{skb} + t_{pkb}) + \mu(\tau_1 + \tau_2 + \frac{6x10^{11}}{\mathcal{B}} + t_{enc+dec})}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{5.13}$$

$$L_q \leq \frac{2\lambda^2(t_{skb} + t_{pkb}) + \lambda\mu(\tau_1 + \tau_2 + \frac{6x10^{11}}{\mathcal{B}} + t_{enc+dec})}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{5.14}$$

(iii) *More than 1 Gbps:* File operation time is negligible compared to the time required to update the group key. We focus on the key update process and use a single server queue which is used to model join-PBC operation. The service time is considered as the time required for the join-PBC operation plus file encryption/decryption time. The service time is expressed as:

$$E[S] = 2\frac{\lambda}{\mu}(t_{skb} + t_{pkb}) + L_q(t_{pkv} + t_{pkb} + t_{skb} + t_{skv}) + \tau_1 + \tau_2 + t_{enc+dec} \tag{5.15}$$

By making the same derivations in the previous case, we derive an upper bound for the performance metrics:

$$W_q \leq \frac{2\lambda(t_{skb} + t_{pkb}) + \mu(\tau_1 + \tau_2 + t_{enc+dec})}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{5.16}$$

$$L_q \leq \frac{2\lambda^2(t_{skb} + t_{pkb}) + \lambda\mu(\tau_1 + \tau_2 + t_{enc+dec})}{\mu - \lambda\mu(t_{pkv} + t_{pkb} + t_{skb} + t_{skv})} \tag{5.17}$$

### 5.3. Numerical Results for the PFSS Use-Case Scenario

In this subsection, we obtain numerical performance results for PFSS based on the healthcare scenario. We consider the average file size as 300 MB. According to Ermis *et al.*, the encryption/decryption time increases linearly with the size of the file in PFSS [11]. Moreover, it takes 3772 miliseconds to encrypt/decrypt a 100 MB file and 8756 miliseconds to encrypt/decrypt a 500 MB file on the platform specified in Section 4.5. Since the encryption/decryption time has linear relation to the size of a file, the encryption/decryption process would consume approximately $(3772 + 8756)/2 = 6264$ miliseconds. Therefore, we conclude that $t_{enc+dec} = 2 * 6264 = 12528$ miliseconds. By using this approximation and execution times reported in Table 4.2, we derive the numerical results for performance metrics under three different bandwidth conditions:

(i) *The Internet connection bandwidth is less than 10 Mbps:* In this case, the performance metrics are as follows:

$$W = \frac{\lambda}{2\mu(\mu - \lambda)} + \frac{1}{\mu} \tag{5.18}$$

$$L = \frac{\lambda^2}{2\mu(\mu - \lambda)} + \frac{\lambda}{\mu} \tag{5.19}$$

where $\lambda$ is the participant arrival rate, $\mu = \frac{\mathcal{B}}{6x10^{11}}$ and $\mathcal{B}$ is the Internet connection bandwidth.

(ii) *The Internet connection bandwidth is between 10 Mbps and 1 Gbps:* In this case, the upper bounds for performance metrics are as follows:

$$W_q \leq \frac{2\lambda(2+1) + \mu(601 + 630 + \frac{6x10^{11}}{\mathcal{B}} + 12528)}{\mu - \lambda\mu(6+1+2+20)} \tag{5.20}$$

$$W_q \leq \frac{6\lambda + \mu(13759 + \frac{6x10^{11}}{\mathcal{B}})}{\mu - 29\lambda\mu} \tag{5.21}$$

$$L_q \leq \frac{2\lambda^2(2+1) + \lambda\mu(601 + 630 + \frac{6x10^{11}}{\mathcal{B}} + 12528)}{\mu - \lambda\mu(6+1+2+20)} \tag{5.22}$$

$$L_q \leq \frac{6\lambda^2 + \lambda\mu(13759 + \frac{6x10^{11}}{\mathcal{B}})}{\mu - 29\lambda\mu} \tag{5.23}$$

(iii) *The Internet connection bandwidth is more than 1 Gbps:* In this case, upper bounds for performance metrics are as follows:

$$W_q \leq \frac{2\lambda(2+1) + \mu(601 + 630 + 12528)}{\mu - \lambda\mu(6+1+2+20)} \tag{5.24}$$

$$W_q \leq \frac{6\lambda + 13759\mu}{\mu - 29\lambda\mu} \tag{5.25}$$

$$L_q \leq \frac{2\lambda^2(2+1) + \lambda\mu(601 + 630 + 12528)}{\mu - \lambda\mu(6+1+2+20)} \tag{5.26}$$

$$L_q \leq \frac{6\lambda^2 + 13759\lambda\mu}{\mu - 29\lambda\mu} \tag{5.27}$$

The average waiting time of the join operation with respect to the participant arrival rate are shown in Figure 5.2. We have analyzed the average waiting times for three different mean residence values: 1 minute, 10 minutes and 50 minutes. In our scenario, PFSS is used as a file sharing system, where medical personnel share records of patients among each other. We assume that each participant has a 1 Gbps Internet connection bandwidth and the average waiting time for a participant before joining the group is between 20 and 60 seconds. If we consider the waiting time threshold as 20 seconds then, the limiting participant rates for each curve are 33 participants, 149 participants and 664 participants at every 100 seconds, respectively. If we consider the waiting time threshold of 60 seconds, the limiting participant rates for each curve are 235 participants, 866 participants and 2202 participants at every 100 seconds, respectively. Therefore, we conclude that PFSS can be used in applications where the

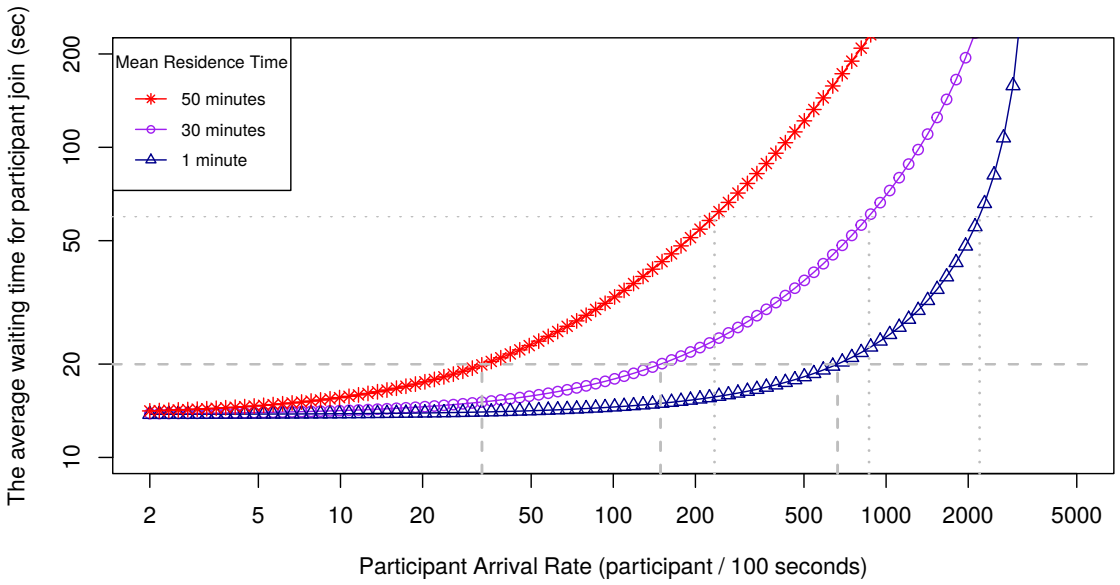



Figure 5.2. Representation of the Average Waiting Time in PFSS with respect to Participant Arrival Rate for different Mean Residence Time

participant arrival rate is high and the mean residence time is less than a few minutes. In such applications, the average waiting time for a joining participant does not exceed 20 milliseconds as long as the participant arrival rate is less than 10 people per second, which is very high for a file sharing application. However, the average waiting time becomes significantly long if PFSS is used in an application with frequent arrival rate and long mean residence time.

We also present a graphical representation regarding the effect of participant bandwidth on the average waiting time in PFSS, as shown in Figure 5.3. We fix the mean residence time to 15 minutes and plot the average waiting time of the join operation with respect to the participant arrival rate for three different Internet connection bandwidth values: 5 Mbps, 50 Mbps and 1 Gbps. As shown in Figure 5.3, the average waiting time is significantly longer when the bandwidth is 5 Mbps. In this case, file upload and download operations become the bottleneck. Even at a participant arrival rate of 100 participants at every 10000 seconds, the system becomes unavailable. Therefore, we deduce that PFSS should not be used in a file sharing application where the average file size is high but the Internet connection bandwidth of participants is

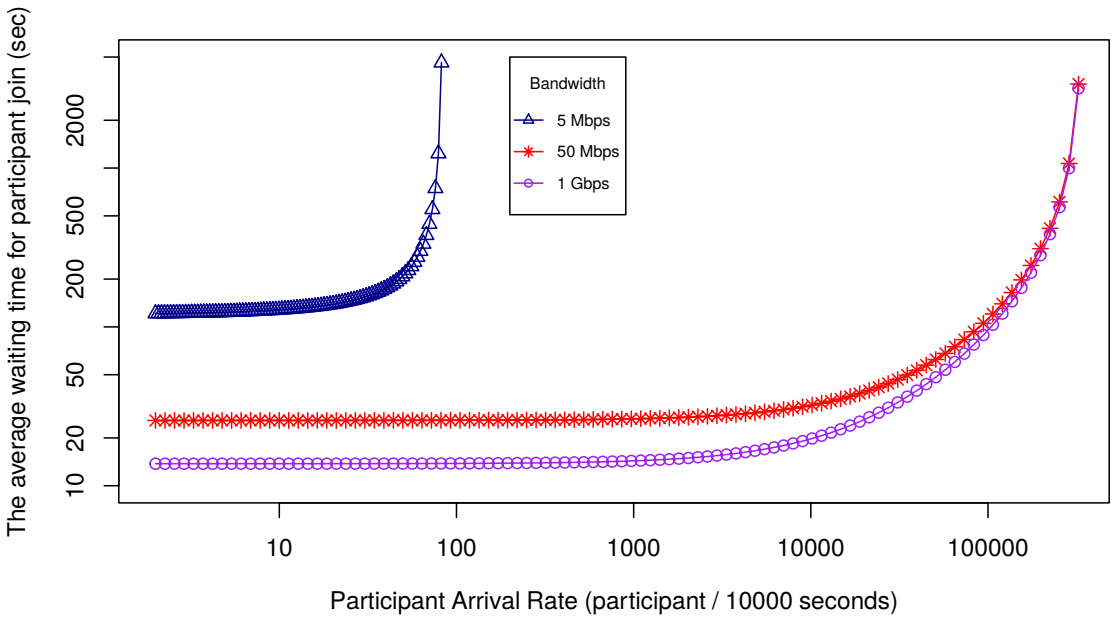



Figure 5.3. Representation of the Average Waiting Time in PFSS with respect to Participant Arrival Rate for different Internet Connection Bandwidth

low. When the bandwidth is 50 Mbps or 1 Gbps, results are similar. The difference between the average waiting time of the two cases is more visible when the participant arrival rate is low. When the participant arrival rate is higher, the number of participants in the group increases and the file transmission time becomes negligible compared to the group key update time. Eventually, the average waiting time of the two cases converges when the participant arrival rate is approximately 10 participants per second.

# 6. CONCLUSION

In this thesis, we have proposed an analytical model for the scalability of dynamic key agreement protocols. First, we have modelled the join and the mass join operations by using networks of queues. Then, we have presented an illustration for the use of our model on an example group key agreement protocol, namely KAP-PBC [11]. In addition, we have derived the average waiting time for joining the group in terms of the participant arrival rate and the computational time complexity of cryptographic operations. Our performance models can help group key agreement protocol designers to estimate the average waiting time of joining participants with respect to the group residence time of participants. Moreover, we have proposed a model for the average waiting time of leaving participants. Furthermore, we have deduced that KAP-PBC could be used in applications with short residence time or low participant arrival rates such as one new participant at every 1000 seconds.

In order to show the applicability of our model on a real life scenario, we have extended the model for secure file sharing systems. For a demonstrative example, PFSS in [11] is employed as an health-care system scenario to securely share records of patients among medical personnel. We perform a numerical analysis on the scenario and conclude that PFSS should be used in applications where the participant arrival rate is high and the mean residence time is less than a few minutes. In addition, we obtain the average waiting time for joining participants with respect to the participant arrival rate for different bandwidth capacity values. We observe that using PFSS in applications where the average file is large such as 500 MB and the participants have low bandwidth capacity such as 5 Mbps may violate the system availability. These results can be used for estimating the performance of a file sharing system. In addition, by adjusting the average file size and bandwidth parameters, our performance models can be applied to various applications. File sharing system admins can utilize these models to estimate the time required for the re-encryption process of a shared file.

## 6.1. Future Work

Future directions to improve the research presented in this thesis are as follows:

- We have used formal modelling and analysis to derive the performance metrics of the proposed model. Therefore, simulation of our performance model in order to validate our theoretical results is left as a future work.
- We have designed three attack models that can be used against dynamic group key agreement protocol based communication systems. In the future, we are planning to simulate these attack models on an example communication system to show how performance metrics are affected due to an attack.
- We have demonstrated a use case scenario to show how our performance model can be used as a healthcare system that is used for storing and sharing medical records of patients. In the future, we are planning to study on different scenarios to analyze the use of group key agreement protocols in different applications.
- We are also planning to employ well-known file size distributions into our model to build a more general performance model for various types of secure file sharing systems.

# REFERENCES


1. Diffie, W. and M. E. Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, Vol. 22, pp. 644–654, 1976.

2. Ingemarsson, I., D. T. Tang, and C. K. Wong, "A Conference Key Distribution System", *IEEE Transactions on Information Theory*, Vol. 28, pp. 714–719, 1982.

3. Burmester, M. and Y. Desmedt, "A Secure and Efficient Conference Key Distribution System (Extended Abstract)", *EUROCRYPT*, 1994.

4. Tzeng, W.-G., "A Secure Fault-Tolerant Conference-Key Agreement Protocol", *IEEE Transactions on Computers*, Vol. 51, pp. 373–379, 2002.

5. Diffie, W., P. C. van Oorschot, and M. J. Wiener, "Authentication and Authenticated Key Exchanges", *Des. Codes Cryptography*, Vol. 2, pp. 107–125, 1992.

6. Tseng, Y.-M., "An Improved Conference-Key Agreement Protocol with Forward Secrecy", *Informatica, Lith. Acad. Sci.*, Vol. 16, pp. 275–284, 2005.

7. Zhao, J., D. Gu, and Y. Li, "An Efficient fault-tolerant group key agreement protocol", *Computer Communications*, Vol. 33, pp. 890–895, 2010.

8. Tzeng, W.-G., "A Practical and Secure Fault-Tolerant Conference-Key Agreement Protocol", *Public Key Cryptography*, 2000.

9. Huang, K.-H., Y.-F. Chung, H.-H. Lee, F. Lai, and T.-S. Chen, "A Conference Key Agreement Protocol with Fault-Tolerant Capability", *Computer Standards and Interfaces*, Vol. 31, pp. 401–405, 2009.

10. Tseng, Y.-M., "A communication-efficient and fault-tolerant conference-key agreement protocol with forward secrecy", *The Journal of Systems and Software*, Vol. 80, pp. 1091–1101, 2007.

11. Ermiş, O., Şerif Bahtiyar, E. Anarım, and M. U. Çağlayan, "A key agreement protocol with partial backward confidentiality", *Computer Networks*, Vol. 129, No. Part 1, pp. 159 – 177, 2017.

12. Cheng, Z.-Y., Y. Liu, C.-C. Chang, and C. Guo, "A fault-tolerant group key agreement protocol exploiting dynamic setting", *International Journal of Communication Systems*, Vol. 26, pp. 259–275, 2013.

13. Sundarapandian, V., "Queueing Theory", *Probability, Statistics and Queueing Theory*, chapter 7, PHI Learning, 2009.

14. Kingman, J. F., "The First Erlang Century–and the Next", *Queueing Syst. Theory Appl.*, Vol. 63, No. 1-4, pp. 3–12, December 2009.

15. Bhat, U. N., *The General Queue G/G/1 and Approximations*, pp. 169–183, Birkhäuser Boston, Boston, 2008.

16. Kingman, J. F. C., "The single server queue in heavy traffic", *Mathematical Proceedings of the Cambridge Philosophical Society*, Vol. 57, No. 4, p. 902–904, 1961.

17. Bekker, R., G. M. Koole, B. F. Nielsen, and T. B. Nielsen, "Queues with waiting time dependent service", *Queueing Systems*, Vol. 68, No. 1, pp. 61–78, May 2011.

18. Abouee-Mehrizi, H. and O. Baron, "State-dependent M/G/1 Queueing Systems", *Queueing Syst. Theory Appl.*, Vol. 82, No. 1-2, pp. 121–148, February 2016.

19. Little, J. D. C. and S. C. Graves, *Little's Law*, pp. 81–100, Springer US, Boston, MA, 2008.

20. Banik, A. D., "Analyzing State-dependent Arrival in GI/BMSP/1/$\infty$ Queues", *Math. Comput. Model.*, Vol. 53, No. 5-6, pp. 1229–1246, March 2011.

21. Sabri Laghaie, K., M. Saidi Mehrabad, and A. Motaghedi Larijani, "Optimization Models for a Deteriorating Single Server Queuing Production System", *Interna-*

*tional Journal of Industiral Engineering & Producion Research*, Vol. 22, No. 4, pp. 251–260, 2011.

22. Wu, T. C., "Conference key distribution system with user anonymity based on algebraic approach", *IEE Proceedings - Computers and Digital Techniques*, Vol. 144, No. 2, pp. 145–148, March 1997.

23. Cheng, J.-C. and C.-S. Laih, "Conference key agreement protocol with non-interactive fault-tolerance over broadcast network", *International Journal of Information Security*, Vol. 8, p. 1, 2009.

24. Wu, Q., Y. Mu, W. Susilo, B. Qin, and J. Domingo-Ferrer, "Asymmetric Group Key Agreement", Joux, A. (editor), *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, Vol. 5479 of *Lecture Notes in Computer Science*, pp. 153–170, Springer, 2009.

25. Tan, Y., J. Zheng, Q. Zhang, X. Zhang, Y. Li, and Q. Zhang, "A Specific-Targeting Asymmetric Group Key Agreement for Cloud Computing", *Chinese Journal of Electronics*, Vol. 27, No. 4, pp. 866–872, 2018.

26. Chuang, Y.-H. and Y.-M. Tseng, "An efficient dynamic group key agreement protocol for imbalanced wireless networks", *International Journal of Network Management*, Vol. 20, pp. 167–180, 2010.

27. Bresson, E., O. Chevassut, A. Essiari, and D. Pointcheval, "Mutual Authentication and Group Key Agreement for Low-power Mobile Devices", *Comput. Commun.*, Vol. 27, No. 17, pp. 1730–1737, November 2004.

28. Xu, C., X. Huang, M. Ma, and H. Bao, "GAKAV: Group Authentication and Key Agreement for LTE/LTE-A Vehicular Networks", *2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on*

*Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 412–418, December 2017.

29. Qikun, Z., G. Yong, Z. Quanxin, W. Ruifang, and T. Yu-An, "A Dynamic and Cross-Domain Authentication Asymmetric Group Key Agreement in Telemedicine Application", *IEEE Access*, Vol. 6, pp. 24064–24074, 2018.

30. Ermiş, O., c. Bahtiyar, E. Anarım, and M. U. Çağlayan, "An Improved Fault-tolerant Conference-key Agreement Protocol with Forward Secrecy", *Proceedings of the 6th International Conference on Security of Information and Networks*, SIN '13, pp. 306–310, ACM, New York, NY, USA, 2013.

31. Lee, S., J. Kim, and S. J. Hong, "Security weakness of Tseng's fault-tolerant conference key agreement protocol", *The Journal of Systems and Software*, Vol. 82, pp. 1163–1167, 2009.

32. Amazon.com Inc., *Amazon Simple Storage Service*, 2019, `https://aws.amazon.com/s3/`, accessed at January 2019.

33. Microsoft Corporation, *Microsoft One Drive*, 2019, `https://account.microsoft.com/account/onedrive`, accessed at January 2019.

34. Google LLC, *Google Drive*, 2019, `https://www.google.com/drive/`, accessed at January 2019.

35. Apple Inc., *iCloud*, 2019, `https://www.icloud.com/`, accessed at January 2019.

36. DropBox Inc., *Dropbox*, 2019, `https://www.dropbox.com/`, accessed at January 2019.

37. Gong, L., "Enclaves: enabling secure collaboration over the Internet", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 3, pp. 567–575, April 1997.

38. Liu, X., Y. Zhang, B. Wang, and J. Yan, "Mona: Secure Multi-Owner Data Sharing

for Dynamic Groups in the Cloud", *IEEE Trans. Parallel Distrib. Syst.*, Vol. 24, No. 6, pp. 1182–1191, June 2013.

39. Malarvizhi, M., J. Sujana, and T. Revathi, "Secure File Sharing Using Cryptographic Techniques in Cloud", *Green Computing Communication and Electrical Engineering (ICGCCEE), 2014 International Conference on*, pp. 1–6, March 2014.

40. Zhu, Z. and R. Jiang, "A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud", *IEEE Trans. Parallel Distrib. Syst.*, Vol. 27, No. 1, pp. 40–50, January 2016.

41. Sharan, P. L. and N. G. Shrikanth, "Secure file sharing technique on cloud storage using an aggregate-key", *National Conference on Advanced Innovation in Engineering and Technology*, NCAIET '15, pp. 269–271, IJIREEICE, India, 2015.

42. Xiong, H., X. Zhang, D. Yao, X. Wu, and Y. Wen, "Towards End-to-end Secure Content Storage and Delivery with Public Cloud", *Proceedings of the Second ACM Conference on Data and Application Security and Privacy*, CODASPY '12, pp. 257–266, ACM, New York, NY, USA, 2012.

43. Wang, M., P. P. Jayaraman, R. Ranjan, K. Mitra, M. Zhang, E. Li, S. Khan, M. Pathan, and D. Georgeakopoulos, *An Overview of Cloud Based Content Delivery Networks: Research Dimensions and State-of-the-Art*, pp. 131–158, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

44. Duarte, E., F. Pinheiro, A. Zúquete, and H. Gomes, "Secure and trustworthy file sharing over cloud storage using eID tokens", *CoRR*, Vol. abs/1501.03139, 2015.

45. Tang, Y., P. P. C. Lee, J. C. S. Lui, and R. Perlman, "Secure Overlay Cloud Storage with Access Control and Assured Deletion", *IEEE Trans. Dependable Secur. Comput.*, Vol. 9, No. 6, pp. 903–916, November 2012.

46. Yamai, N., K. Nakayoshi, H. Ishibashi, K. Abe, K. Murakami, and T. Matsuura,

"NFS-based secure file sharing over multiple administrative domains with minimal administration", *Systems and Computers in Japan*, Vol. 33, No. 14, pp. 50–58, 2002.

47. Yu, S., C. Wang, K. Ren, and W. Lou, "Attribute Based Data Sharing with Attribute Revocation", *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '10, pp. 261–270, ACM, New York, NY, USA, 2010.

48. Bethencourt, J., A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption", *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, SP '07, pp. 321–334, IEEE Computer Society, Washington, DC, USA, 2007.

49. Blaze, M., G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography", Nyberg, K. (editor), *Advances in Cryptology — EUROCRYPT'98*, Vol. 1403 of *Lecture Notes in Computer Science*, pp. 127–144, Springer Berlin Heidelberg, Helsinki, Finland, 1998.

50. Wang, G., Q. Liu, J. Wu, and M. Guo, "Hierarchical Attribute-based Encryption and Scalable User Revocation for Sharing Data in Cloud Servers", *Comput. Secur.*, Vol. 30, No. 5, pp. 320–331, July 2011.

51. Boneh, D., X. Boyen, and E.-J. Goh, "Hierarchical Identity Based Encryption with Constant Size Ciphertext", Cramer, R. (editor), *Advances in Cryptology – EUROCRYPT 2005*, pp. 440–456, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

52. Waters, B., *Public Key Cryptography – PKC 2011: 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings*, chapter Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization, pp. 53–70, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

53. Backes, M., C. Cachin, and A. Oprea, "Lazy revocation in cryptographic file systems", *Third IEEE International Security in Storage Workshop (SISW'05)*, pp. 11 pp.–11, IEEE, San Francisco, CA, USA, December 2005.

54. Liu, Q., G. Wang, and J. Wu, "Time-based Proxy Re-encryption Scheme for Secure Data Sharing in a Cloud Environment", *Inf. Sci.*, Vol. 258, pp. 355–370, February 2014.

55. Abdel-Harfez, A., A. Miri, and L. Orozco-Barbosa, "Authenticated Group Key Agreement Protocols for Ad hoc Wireless Networks.", *I. J. Network Security*, Vol. 4, pp. 90–98, 2007.

56. Augot, D., R. Bhaskar, V. Issarny, and D. Sacchetti, "A three round authenticated group key agreement protocol for ad hoc networks", *Pervasive and Mobile Computing*, Vol. 3, pp. 36–52, 2007.

57. Chang, C.-C., H.-C. Tsai, P.-Y. C. L. Chu, X. Zheng, and S. Wang, "A Collaborative Conference Key Agreement Scheme by Using an Intermediary Node", *Proceedings of the 2007 International Conference on Convergence Information Technology*, 2007.

58. Chung, Y.-F., "The design of authentication key protocol in certificate-free public key cryptosystem", *Security and Communication Networks*, Vol. Security Communication Networks, 2013; doi: 10.1002/sec.924, 2013.

59. Ermiş, O., c. Bahtiyar, E. Anarim, and M. U. Çağlayan, "A Secure and Efficient Group Key Agreement Approach for Mobile Ad Hoc Networks", *Ad Hoc Netw.*, Vol. 67, No. C, pp. 24–39, December 2017.

60. Gangwar, R. C. and A. K. Sarje, "Secure and Efficient Dynamic Group Key Agreement Protocol for an Ad Hoc Network", *Ad Hoc and Ubiquitous Computing, 2006. ISAUHC '06. International Symposium on*, 2006.

61. Li, M., X. Xu, C. Guo, and X. Tan, "AD-ASGKA – authenticated dynamic protocols for asymmetric group key agreement", *Security and Communication Networks*, Vol. 9, No. 11, pp. 1340–1352, 2016, sCN-15-0041.R1.

62. Teng, J. and C. Wu, "An Identity-Based Group Key Agreement Protocol for Low-Power Mobile Devices", *Chinese Journal of Electronics*, Vol. 25, No. 4, pp. 726–733, 2016.

63. Zhang, L., Q. Wu, J. Domingo-Ferrer, B. Qin, and Z. Dong, "Round-Efficient and Sender-Unrestricted Dynamic Group Key Agreement Protocol for Secure Group Communications", *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 11, pp. 2352–2364, November 2015.

64. Ermiş, O., c. Bahtiyar, E. Anarim, and M. U. Çağlayan, "A Comparative Study on the Scalability of Dynamic Group Key Agreement Protocols", *Proceedings of the 12th International Conference on Availability, Reliability and Security*, ARES '17, pp. 62:1–62:6, ACM, New York, NY, USA, 2017.

65. Choi, C. H., "Adoptation of Weil Pairing IBE for Secure File Sharing", *The Fifth International Conference on Advances in Databases, Knowledge, and Data Applications*, GlobeNet 2013, pp. 59–65, IARIA, Seville, Spain, 2013.

66. Huang, K. H., E. C. Chang, and C. L. Chang, "Secure File Sharing Scheme for Mobile Devices", *2013 Fourth International Conference on Networking and Distributed Computing*, pp. 82–84, December 2013.

67. Ibraimi, L., M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated Ciphertext-Policy Attribute-Based Encryption and Its Application", Youm, H. Y. and M. Yung (editors), *Information Security Applications*, pp. 309–323, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

68. Newell, G. F., "The $M/G/\infty$ Queue", *SIAM Journal on Applied Mathematics*, Vol. 14, No. 1, pp. 86–88, 1966.

69. Menasce, D. A., V. A. Almeida, and L. W. Dowdy, *Performance by Design: Computer Capacity Planning by Example*, Prentice-Hall, 2004.

70. Chaudhry, M. L. and J. G. C. Templeton, *A First Course in Bulk Queues*, Wiley, New York, 1983.

71. Murthy, T. S. R., S. Krishna, and G. Raju, "Interdependent Queueing Model with Varying Bulk Service", *International Journal of Mathematics and Soft Computing*, Vol. 2, No. 1, pp. 109–117, 2012.

72. The Ohio State University, Mathematics Department, *Infinite Series*, 2019, `https://people.math.osu.edu/husen.1/teaching/530/series.pdf`, accessed at January 2019.

73. Downey, A. B., "The Structural Cause of File Size Distributions", *SIGMETRICS Perform. Eval. Rev.*, Vol. 29, No. 1, pp. 328–329, June 2001.

74. Matsumoto, T., T. Onoyama, and N. Komoda, *Efficient Operational Management of Enterprise File Server with File Size Distribution Model*, pp. 599–609, Springer Netherlands, Dordrecht, 2014.

75. Mitzenmacher, M., "Dynamic Models for File Sizes and Double Pareto Distributions", *Internet Mathematics*, Vol. 1, No. 3, pp. 305–333, 2003.

76. Gonçalves, G., I. Drago, A. P. C. d. Silva, A. B. Vieira, and J. M. Almeida, "Modeling the Dropbox client behavior", *2014 IEEE International Conference on Communications (ICC)*, pp. 1332–1337, IEEE, Sydney, NSW, Australia, June 2014.

77. Saibert, Anthony J., *Archiving: Fundamentals of Storage Technology*, 2019, `http://siim.org/page/archiving_chapter2`, accessed at January 2019.