Design and Analysis of QoS-Aware Key Management and Intrusion Detection Protocols for Secure Mobile Group Communications in Wireless Networks

by

Jin-Hee Cho

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science and Applications

November 12, 2008

Falls Church, Virginia

Keywords: QoS-awareness, group key management, intrusion detection, forward secrecy, backward secrecy, batch rekeying, secure group communications, mobile ad hoc networks, wireless networks, mean time to security failure, performance analysis.

Design and Analysis of QoS-Aware Key Management and Intrusion Detection Protocols for Secure Mobile Group Communications in Wireless Networks

Jin-Hee Cho

ABSTRACT

Many mobile applications in wireless networks such as military battlefield, emergency response, mobile commerce, online gaming, and collaborative work are based on the notion of group communications. Designing security protocols for secure group communications in wireless networks faces many technical challenges due to unique characteristics of wireless networks including resource-constrained environments in bandwidth, memory size, battery life and computational power, openness to eavesdropping and security threats, and unreliable communication. Further, for mobile ad hoc networks (MANETs) with no infrastructure support, rapid changes in network topology due to user mobility could cause group merge/partition events to occur dynamically.

While satisfying security requirements is crucial for secure group communications in wireless systems, mobile group applications often have application-specific performance requirements in terms of timeliness, reliability, and system reconfigurability. Often there exists a tradeoff between security versus performance goals since security protocols may introduce undue computational and network overheads which may prevent performance goals from being met.

Unlike traditional security protocols which concern security properties only, in this dissertation research we design and analyze a class of QoS-aware protocols for secure group communications in wireless networks with the goal to satisfy not only security requirements in terms of secrecy, confidentiality, authentication, availability, and data integrity, but also performance requirements in terms of latency, network traffic, response time, and reconfigurability for secure group communication systems (GCSs) in wireless networks. These QoS-aware protocols are adaptive in nature with designs to allow the system to dynamically adjust operational settings, under which both the system's security and performance requirements can be best satisfied, leveraging the inherent tradeoff between performance versus security goals.

Our contribution has two elements: design and analysis. While our designs mostly derive from existing work, the optimization design principles developed are new to secure GCSs. The analysis methodology developed for the tradeoff analysis of performance versus security of secure group communication protocols is a major contribution. Specifically, the dissertation research has three contributions. First, we propose and analyze efficient, QoS-aware key management protocols for secure group communications in wireless networks to deal with *outsider attacks*. In order to efficiently reduce the network communication cost caused by rekeying operations (i.e., change a group key), three "threshold-based" periodic batch rekeying protocols are proposed and analyzed. The aim of these protocols is to satisfy application security requirements while minimizing the network communication cost. Instead of individual rekeying, i.e., performing a rekeying operation right after each group join or leave request, these protocols perform batch rekeying periodically. We demonstrate that an optimal rekey interval exists for each protocol that would satisfy an imposed security requirement while minimizing the network communication cost. We further compare these protocols against individual rekeying to identify the best protocol that can minimize the communication cost of rekeying while satisfying application requirements, when given a set of parameter values characterizing the operational and environmental conditions of the system.

Second, we propose and analyze QoS-aware intrusion detection protocols for secure group communications in wireless networks to deal with *insider attacks*. These protocols explore the tradeoff of security versus performance properties with the goal to determine the best periodic interval for performing intrusion detection. Specifically, we consider a class of intrusion detection protocols including host-based and voting-based IDS protocols for detecting and evicting compromised nodes and examine their effect on *MTTSF* versus the response time performance metric. Our analysis reveals that there exists an optimal intrusion detection interval under which the *MTTSF* metric can be best traded off for the response time performance metric, or vice versa. Furthermore, the intrusion detection interval can be dynamically adjusted based on the attacker behaviors to maximize *MTTSF* while satisfying a system-imposed response time requirement.

Third, we propose and analyze a scalable and efficient region-based group key management protocol for managing mobile groups in MANETs. For scalability and dynamic reconfigurability, we take a region-based approach by which group members are broken into region-based subgroups, and leaders in subgroups securely communicate with each other to agree on a group key in response to membership change and member mobility events. This key management protocol is proposed to identify the optimal regional area size that minimizes the network communication cost while satisfying the application security requirements. Further, it allows mobile groups to react to network partition/merge events for reconfigurability and survivability while still maintaining the design goal of secure group communications in MANETs. Using the proposed region-based group key management, we identify the optimal regional area size that efficiently trades inter-regional communication overhead off for intra-regional communication overhead. We demonstrate its efficiency by comparing it with a no-region GCS, under a set of identified design parameters characterizing network environments and operational conditions of the targeted application. We further investigate the effect of integrating QoS-aware intrusion detection with region-based group key management in MANETs and identify combined optimal settings in terms of the optimal regional size and the optimal intrusion detection interval under which the security and performance properties of the system can be best optimized.

We evaluate the merits of our proposed QoS-aware security protocols for mobile group communications through model-based mathematical analyses with extensive simulation validation. We perform thorough comparative analyses against baseline secure group communication protocols which do not consider security versus performance tradeoffs, including those based on individual rekeying, no intrusion detection, and/or no-region designs. The results obtained show that our proposed QoS-aware security protocols outperform these baseline algorithms.

To my husband, Chung Won Ha, and my son, Jinwoo Heero Ha

ACKNOWLEDGEMENTS

First of all, I thank God for me to finish this long journey safely. I could fully enjoy this journey he planned because of his protection and guidance.

My best appreciation goes to my advisor, Dr. Ing-Ray Chen. I was privileged to have him as my advisor and received his guidance and support during the long journey of my Ph.D. study. I could really enjoy my Ph.D. study because of his patience, understanding, guidance, generosity, and support so that I could survive as a Ph.D. student while maintaining my family work. I always hoped to absorb his scholarship as well as his character as my role model during my Ph.D. study. Dr. Chen was the first professor I met at Virginia Tech and I hope to have him as my forever mentor who affects my entire life.

I thank my committee members for their support and help. Dr. Scott F. Midkiff helped provide the IREAN (Integrated Research and Education in Advanced Networking) Fellowship and gave me an opportunity to enjoy working with him in the CS/ECE4570 Wireless Networks and Mobile Systems course as a GTA (Graduate Teaching Assistant). I also would like to express my gratitude to Dr. Mohamed Eltoweissy. Dr. Eltoweissy advised me to generate quality work and was willing to spend time to discuss my Ph.D. research issues even during his busy schedule. Dr. Chang-Tien Lu always kindly guided me in a general way and provided me a pleasant work environment by allowing me to share his lab with his students. Also I express thankfulness to Dr. Csaba J. Egyhazy for his critical comments on my Ph.D. dissertation for further improvement.

Last, but not least, I sincerely express my appreciation to my husband, Chung W. Ha, for his endless love and constructive advices to pursue my professional goals. Without his love and patience, I could not finish my Ph.D. work. I would also like to thank my father, Sung-Kyung Cho, and my mother, Kwang-Sook Han, for their endless patience and support. I also thank my mother-in-law, Young-Hee Seo, for her daily support and patience for me to handle both study and family. My great thank also goes to my son, Jinwoo H. Ha. He is the source of my energy.

I also could not forget great time spent with my friends and their help at Virginia Tech, Weiping He, Baoshan Gu, David (Jing) Dai, and Feng Chen. I also would like to express my thanks to Ms. Cynthia D. Robinson for her general support. Further, I express many thanks to my friends, Sung-Won Park, Young-Jun Lee, Jeung-Wha Im, and Kyung-Ho Ahn.

Table of Contents

List of Figures

List of Tables

Chapter 1 INTRODUCTION

1. 1 QoS-Aware Security Protocols for Mobile Group Communications in Wireless Networks

Many mobile applications in wireless networks such as military battlefield, emergency response, mobile commerce, online gaming, and collaborative work are based on the notion of group communications. Designing security protocols for secure group communication systems (GCSs) in wireless networks faces many technical challenges due to unique characteristics of wireless networks including resource-constrained environments in bandwidth, memory size, battery life, and computational power, openness to eavesdropping and security threats, and unreliable communication. Further, for mobile ad hoc networks (MANETs) with no infrastructure support, rapid changes in network topology due to user mobility could cause group merge/partition events to occur dynamically.

To deal with outsider attacks, one way to achieve cost-effective secure GCSs is to use a symmetric key, called the *group key*, shared by group members. The group key may be distributed by a key server that provides group key management services. A dedicated key server may be employed, or the functionality may be implemented on a server offering other services such as authentication. Multiple key servers may co-exist in a clustered network, where a cluster head may play the role of a key server [110]. The group key is employed to encrypt messages sent by a member to the group. Only members of the group with the group key are capable of decrypting the messages [60]. Key generation along with key distribution has been a central issue in key management for secure group communications. Over the years many key management protocols have been proposed and studied (see Chapter 2 Related Work). In particular, in MANETs with no infrastructure support, since a key server does not exist, key management must be performed in a fully distributed manner. This adds to the system overhead whenever the group key is "rekeyed" due to group member leave/join/eviction events. To deal with insiders attacks, intrusion detection system (IDS) techniques may be used to detect compromised nodes and to evict such compromised nodes to prolong the lifetime of the GCS.

While satisfying security requirements is crucial for secure GCSs in wireless systems, mobile group communicating applications often have application-specific performance requirements in terms of timeliness, throughput, delay, and traffic capacity. These application requirements are generally referred to as the quality of service (QoS) requirements, including both security and performance requirements in the context of mobile GCSs. By "QoS-aware" protocols, we refer to those protocols being designed to satisfy both security and performance requirements of the system. These QoS-aware protocols are adaptive in nature with designs incorporated to allow the system to adapt to dynamic situations by adjusting operational settings under which both the system's security and performance requirements can be best satisfied.

1. 2 Research Motivation and Problem Definition

Our research motivation derives from the observation that there often exists a tradeoff between security versus performance goals of GCSs in wireless networks since security protocols may introduce undue computational and network overheads which may prevent performance goals from being met. Unlike traditional security protocols which concern security properties only, in this dissertation research we design and analyze a class of QoS-aware protocols for secure GCSs in wireless networks with the goal to satisfy not only security requirements in terms of secrecy, confidentiality, authentication, availability, and data integrity, but also performance requirements in terms of delay, network traffic, and response time for secure GCSs in wireless networks.

In the literature (see the literature survey in Chapter 2), most existing work focuses on "security" aspects of proposed protocols, without paying much attention to "performance" aspects of the system, particularly for wireless networks with or without infrastructure where resources are scarce and maintaining security is challenging due to unique characteristics of wireless networks. Also, while qualitative studies for network security protocols have been attempted, evaluating security property of the system using model-based quantitative techniques is still in its infancy. We are motivated to develop model-based quantitative modeling techniques to investigate the tradeoff between security and performance properties of the system. Lastly, while most existing security protocols are designed to deal with specific attacks, no security protocols exist to deal with both outside and inside attackers for GCSs. We are motivated to design and analyze a class of QoS-aware security protocols that can cope with both insider and outsider attacks in secure GCSs, utilizing model-based quantitative techniques to identify optimal design settings under which application-specific QoS requirements can be best satisfied.

The research problem in the dissertation research lies in both “design” and “analysis” of QoS-aware security protocols in wireless networks to satisfy both security and performance properties of the GCSs. We focus on four areas: (1) QoS-aware group key management to prevent outsider attacks as well as to mitigate insider attacks; (2) QoS-aware IDS protocols for mainly coping with insider attacks to prolong the lifetime of GCSs; (3) scalable and reconfigurable group key management for MANETs; and (4) integrated QoS-aware protocol suites to deal with both insider and outsider attacks.

1. 3 Contributions

Our dissertation research aims at the design and analysis of a class of QoS-aware key management and IDS protocols with design features incorporated to allow the system to dynamically adjust operational settings for satisfying the imposed security requirements while maximizing the system performance for supporting secure GCSs in wireless networks. While our protocol designs mostly derive from existing work, the optimization design principles developed are new for secure GCSs. Further, the analysis methodology developed as a general framework for the tradeoff analysis of performance versus security of secure group communication protocols is a major contribution. We consider this issue for a set of wireless networks, including wireless networks with infrastructure support (e.g., a centralized key server exists) and MANETs without infrastructure support. We summarize specific contributions of our dissertation research as follows.

First, we design and analyze a class of QoS-aware “threshold-based” periodic batch rekeying protocols for efficient group key management in wireless networks with the objective to satisfy both the security and performance requirements of secure GCSs. We demonstrate that an optimal *batch rekey interval* exists that would satisfy imposed security requirements while minimizing the network traffic and delay. We further compare these protocols to identify the best protocol that can minimize the communication cost of rekeying while satisfying security and performance QoS requirements, when given a set of design parameter values characterizing the operational and environmental conditions of the system. The optimization design principles developed are demonstrated to be generally applicable to the system in which a centralized server exists, and to the system in which no infrastructure exists (e.g., MANETs) for group key generation and distribution.

Second, we design and analyze a class of QoS-aware IDS protocols employed in wireless networks to deal with insider attacks to satisfy both the security and performance requirements of the system. Specifically, we consider a class of QoS-aware intrusion detection protocols including host-based and voting-based IDS protocols and examine their effect on the system mean time to security failure (*MTTSF*) and overall communication cost performance metrics. Our analysis reveals that there exists an optimal intrusion detection interval under which the *MTTSF* metric can be best traded off for the overall communication cost performance metric, or vice versa. Furthermore, the intrusion detection interval can be dynamically adjusted based on the attacker behaviors detected to maximize *MTTSF* while satisfying a system-imposed acceptable performance requirement. We conduct model-based performance analyses of these proposed QoS-aware IDS protocols in single-hop peer-to-peer MANETs as well as in multi-hop MANETs.

Third, we design and analyze a region-based group key management architecture for secure GCSs in the context of MANETs in which there is no infrastructure support. For scalability and dynamic reconfigurability, we take a region-based approach by which group members are broken into region-based subgroups. Leaders in subgroups securely communicate with each other to agree on a group key in response to membership change and member mobility events. We identify the optimal regional area size under which the system can satisfy imposed security requirements, while minimizing the network traffic and delay.

Fourth, given that there exists no prior work that deals with both insider and outsider attacks for GCSs in MANETs, we integrate QoS-aware IDS and group key management protocols for dealing with both inside and outside attackers for GCSs in MANETs and identify optimal settings under which *MTTSF* is maximized while satisfying performance requirements. This includes integrating threshold-based periodic batch rekeying with QoS-aware IDS techniques and identifying optimal settings in terms of the optimal thresholds used for batch rekeying and the optimal intrusion detection interval. For scalability and dynamic reconfigurability, we also integrate QoS-aware IDS protocols with region-based group key management and identify the effects of intrusion detection interval and regional area size on the security and performance properties of the system.

Fifth, we develop mathematical models based on stochastic Petri nets (SPN) to identify optimization design conditions to be employed at runtime. These models follow the concept of model-based quantitative analysis. They are novel as there is no prior work in mathematical modeling for tradeoff analyses of security versus performance properties of mobile GCSs. Also these mathematical models generically describe the behaviors of a mobile GCS in wireless networks and can be easily extended to reflect changes in operational and environmental conditions, attack behaviors, system failure definitions and workload conditions, thereby allowing optimal settings to be identified by the system to best satisfy security and performance requirements. The validity of these mathematical models is ascertained through extensive simulation in the dissertation research.

1. 4 Dissertation Organization

The rest of this dissertation is organized as follows. Chapter 2 surveys the literature on existing group key management and intrusion detection protocols for secure group communications in wireless networks, and contrasts prior work with our work. Chapter 3 describes our system model and assumptions made for characterizing GCSs in wireless networks, including both infrastructure and infrastructure-less networks. In Chapter 4, we design and analyze a class of QoS-aware threshold-based periodic batch rekeying protocols for achieving efficient key management to mainly deal with outsider attacks in wireless networks where a centralized key server exists and also in MANETs. Chapter 5 reports our research results on the design and analysis of a class of distributed IDS protocols to deal with insider attacks for mobile GCSs in MANETs, including the case in which all nodes are reachable within one-hop radio range and the case of multi-hop environments. In Chapter 6, we integrate QoS-aware IDS with threshold-based periodic batch rekeying to deal with both insider and outsider attacks in multi-hop MANETs. In Chapter 7, we design and analyze region-based group key management protocols for scalable and reconfigurable group key management in MANETs. In Chapter 8, we integrate QoS-aware IDS with region-based group key management for scalability and reconfigurability to deal with both insider and outsider attacks in multi-hop MANETs. Chapter 9 summarizes the dissertation research work, discusses applicability of the proposed QoS-aware security protocols, and outlines some future research areas.

Chapter 2 RELATED WORK

In this chapter, we survey existing work on group key management and intrusion detection protocols for secure group communications in wireless networks. Specifically, Section 2.1 surveys existing work in periodic batch rekeying protocols. Section 2.2 addresses prior work on distributed intrusion detection protocols for MANETs and quantitative modeling studies for intrusion tolerance/detection systems. Section 2.3 reviews distributed key management protocols, especially contributory key agreement (CKA) protocols and group key management protocols for MANETs. We compare and contrast these existing algorithms and point out the novelty of our work with respect to existing work.

2.1 Periodic Batch Rekeying

For large dynamic groups, a join or leave request can occur very frequently. *Individual rekeying* performs a rekeying operation whenever a new user joins the group or a current member leaves the group or is evicted. This is not scalable to a large dynamic group because of the significant communication overhead incurred by frequent rekeying in bandwidth-constrained wireless communications. The overhead is exacerbated by the need to authenticate each rekeying message. Moreover, synchronization is difficult to maintain if the group key is rekeyed immediately after each join or leave [60]. To remedy this, researchers have proposed *periodic batch rekeying* [42, 60, 89, 110] by which join and leave requests are aggregated and rekeying is performed only periodically. A consequence of batch rekeying is that members may not immediately join or leave the group. Thus, forward and backward secrecy requirements may not be strictly satisfied.

Hardjono et al. [42] proposed *periodic batch rekeying* to decrease rekeying overheads in dynamic group communications. Li et al. [60] proposed the use of periodic batch rekeying to improve efficiency and reduce the out-of-sync problem. Setia et al. [111] described an approach for scalable group rekeying for secure multicast using periodic group rekeying, called *Kronos*. They discussed the inefficiency of individual rekeying under dynamic and large networks, and compared the performance of *Kronos* with other key management protocols using simulation.

Yang et al. [108] designed a batch-rekeying algorithm, called *keygem,* to improve scalability and performance of a large and dynamic group. Moharrum et al. [69] proposed a method to handle group dynamics in a multicast key tree and maintain a balanced tree with minimal cost. Recently, Lazos and Poovendran [57] proposed the use of location-aware batch rekeying of key hierarchies in wireless ad hoc networks. However, no prior work dealt with identifying optimal rekeying periods. Our dissertation research is the first work that addresses the issue of optimal batch rekey interval to minimize the communication cost per join/leave operation while satisfying imposed security and performance requirements. Specifically, we develop *threshold-based periodic batch rekeying protocols* and demonstrate that an optimal batch rekey interval exists in each protocol. We compare these protocols to identify the best protocol that can minimize the communication cost for rekeying while satisfying application requirements (i.e., secrecy and delay), when given a set of design parameter values characterizing the operational and environmental conditions of the system. In a highly dynamic wireless environment in which the system design parameter values change at runtime, our work may be used to adapt the batch rekey interval accordingly.

In our work, we investigate the use of the Logical Key Hierarchy (*LKH*) rekeying protocol [74] for the case in which a key server exists in our threshold-based periodic batch rekeying protocols. *LKH* is categorized as one of centralized key management protocols assuming there is a centralized server to deal with various key management tasks [74]. Di Pietro et al. [75] proposed *LKH++* which extends the basic *LKH* protocol [74] to efficiently provide secure group key management for mobile users. Di Pietro et al. [76] also proposed *LKHW*, a scheme that combines *directed-diffusion* and *LKH* for efficient key management in wireless sensor networks. Several other schemes reported in [41] use enhancements of *LKH* for key management in wireless networks. Although *LKH* has been used for key management in wireless networks and periodic batch rekeying has been proposed as an efficient strategy to reduce rekeying overhead by trading secrecy violation off for reducing rekeying overheads, the issue of optimal batch rekey intervals has not been addressed and the relationship between the optimal batch rekey interval and environmental conditions (i.e., the arrival rate of join or leave requests or the probability of trustworthiness in receiving requests) remains to be investigated. Our work addresses this issue.

2.2 Distributed Intrusion Detection Protocols

While intrusion detection systems (IDS) for wired networks have been extensively studied, there has been little work on IDSs for wireless mobile environments, particularly for MANETs. Zhang et al. [112, 113] pioneered a distributed and cooperative host-based intrusion detection model based on anomaly detection by which all nodes in the system run IDS to detect and respond to intrusion. In their model, an IDS agent on each node can perform local data collection and intrusion detection within radio range. Global intrusion detection is achieved by cooperation among neighboring nodes. Nevertheless, no specific IDS protocol was discussed.

Recently, cluster-based IDS for MANETs has been proposed [12, 14, 45, 50, 94, 99, 100]. The main idea is that instead of performing IDS at each node, a cluster head (CH) collects security-related information from nodes in the same cluster and determines if intrusion has occurred. In particular, non-overlapping zone-based IDS was proposed in [99, 100] for MANETs and proven to be effective in intrusion detection. An important issue not addressed is performance degradation due to zone-based IDS. Marti et al. [65] developed a *watchdog* mechanism for identifying misbehaving nodes based on dynamical behaviors and developed a *pathrater* algorithm for routing around misbehaving nodes for MANETs. Debar et al. [34] suggested aggregation and correlation of IDS alerts to reduce communication/computational overhead caused by performing IDS. Hierarchical IDS was proposed in [14, 45, 50] to realize distributed anomaly-based IDS in MANETs. However, the issues of extra latency and energy consumption are not addressed. The assumptions that the CH is tamper-resistant and the CH selection process will not be interrupted by attackers are also questionable. In our dissertation work, we develop voting-based IDS by which multiple nodes participate in the eviction process of a target node and perform a distributed majority voting to determine if the target node is to be evicted. This eliminates a single point of failure problem in cluster-based IDS protocols.

These previous studies cited above only examined security properties of IDS in MANETs, not impact of performance degradation introduced by IDS used. Stern et al. [94] proposed data reduction techniques to reduce communication cost in their IDS design. However, detection latency introduced by data aggregation and their effect on performance degradation were not investigated. Our work differs from prior work in that our protocol is QoS-aware considering the impact of IDS on performance degradation and analyzing the tradeoff between performance (i.e., network communication cost or service response time) versus security (i.e., *MTTSF*) for host-based and voting-based IDS techniques applied to GCSs in MANETs.

Recently, Subhadrabandhu et al. [96, 97, 98] studied the tradeoff between energy, computational, communication resource consumption versus IDS accuracy based on distributed IDS. Algorithms were developed to explore the tradeoff. Our proposed intrusion detection protocols differ from their work in that we specifically deal with GCSs in MANETs. Moreover, our protocols are QoS-aware with optimization designs to identify optimal detection intervals to maximize the system lifetime *MTTSF* while satisfying performance requirements in terms of communication cost and/or response time performance metrics.

Below we survey prior work on model-based intrusion tolerance/detection systems using quantitative modeling techniques since our work has its root in model-based quantitative analysis [71]. In the literature, not much work has been done in extending model-based quantitative analysis to security analysis. Zhang et al. [114] analyzed several group rekeying algorithms in wireless environments and evaluated their performance characteristics. No intrusion was considered, however. Dacier et al. [32] proposed a novel approach to model the system as a privilege graph demonstrating operational security vulnerabilities and transformed the privilege graph into a *Markov chain* based on all possible successful attack scenarios. Jonsson et al. [48] presented a quantitative Markov model of attacker behaviors using data obtained from several experiments conducted over two years. They postulated that the process describing attacker behaviors may be divided into multiple phases, such as learning, standard attack, and innovative attack. Popstojanova et al. [77] presented a state transition model to describe dynamic behaviors of intrusion tolerance systems. Their model includes a framework to define the vulnerability and the threat set. Madan et al. [63, 64] employed a Semi-Markov Process (SMP) model to evaluate security attributes of an intrusion-tolerant system. Based on particular attack scenarios, the states are related with failure of availability, data integrity, and data confidentiality. A steady-state analysis has been used to obtain dependability measures such as availability. A transient analysis with absorbing states has been used to obtain security measures such as *MTTSF* similar to the computation of the *mean time to failure* (*MTTF*) in reliability analysis. Stevens et al. [95] also proposed a networked intrusion tolerant information system using a model-based validation technique, so called probabilistic modeling. Their model-based results were employed not only to guide the system's design but also to determine whether or not a given survivability requirement was met. They used two security metrics: *mean time to discovery* (*MTTD*) referring to the mean time between successive discoveries of unknown vulnerabilities and mean time to exploit (*MTTE*) meaning the mean time between successive exploitations of a known vulnerability. Wang et al. [105] utilized a higher-level formalism based on Stochastic Petri nets (*SPN*) for security analysis of intrusion tolerant systems. Recently, Leversage and James [59] suggested a security metric to intelligently compare systems and to make corporate security decisions. They proposed a *mean time-to-compromise (MTTC)* metric to measure the time needed for an attacker to successfully disrupt a target system. Most previous work cited above, however, often only focused on security measures without considering the impact of deploying security protocols on the performance of the system. We believe that for model-based quantitative analysis, the definitions and designs of security properties should reflect specific network and workload environments and take both security and performance requirements into consideration. In the dissertation work, we develop model-based analysis techniques that consider both security and performance metrics with the objective to identify operational settings under which both security and performance requirements can be best satisfied or optimized.

2.3 Hierarchical Group Key Management Protocols

Group key management can generally be classified into centralized, decentralized, and distributed [39, 80, 111]. A centralized scheme uses a key controller for key management tasks including key generation, assignment, distribution, and revocation and is not suitable for MANETs. A decentralized scheme divides a group into subgroups typically hierarchically to spread out the workload of a central controller. A distributed scheme does not have a group key controller for group key management. Instead, a group key is generated in a contributory manner by all members in the system. In this dissertation research, we develop a region-based group key management protocol, which can be considered as a hybrid of decentralized and distributed schemes. Similar to a decentralized scheme, our region-based group key management protocol divides a group into region-based subgroups in a two-level hierarchy. Similar to a distributed scheme, however, there is no group key controller within each region and all members contribute to key generation and distribution.

Over the past few years, there have been hierarchical group key management protocols proposed in the literature. IGKMP [42, 43, 44, 114] divides a group into several subgroups to enhance scalability. They proposed several rekeying algorithms that preserve secrecy properties as members move within the hierarchy. Within each subgroup, a set of key controllers is used for key management. This approach is suitable for wired networks since key controllers are stationary and wired communication to key controllers is reliable. However, it is not suitable for MANETs where nodes are mobile and wireless communication is often unreliable. Rafaeli et al. proposed *Hydra* [80] that divides a group into a number of TTL (Time-to-Live)-scoped regions for flexible and efficient group key management to support secure multicasting. *Hydra*, however, is also based on the use of multiple group controllers in a region. Dondeti et al. [36] proposed DEP for secure multicasting based on a hierarchical subgrouping architecture for scalability. Again, DEP also uses multiple subgroup controllers. Similarly, *Iolus* [68] is a framework that divides a group into smaller subgroups each with multiple subgroup controllers. In [41], HKT was proposed to balance security and efficiency making use of a two level hybrid key tree based on clusters. Cluster sizes are adjusted depending on the level of collusion resistance. We observe that these hierarchical group key management protocols are designed for wired networks and are not suitable for MANETs.

Most existing work on hierarchical group key management in MANETs considered hierarchical clustering for grouping nodes into clusters for scalability and efficiency [5, 6, 7, 8, 57, 61, 82, 104]. Rhee et al. [80] employed a two-layer hierarchical key management structure for secure group communications overseen by unmanned aerial vehicles (UAVs). They considered the use of a stationary super-node as a cluster head. Bechler et al. [8] proposed an efficient distributed key management based on hierarchical clustering. However, no optimal setting of their proposed protocol was identified to maximize system performance. Work in [7, 57, 104] proposed clustering algorithms in MANETs for energy conservation, with the effort mainly focused on the effect of clustering on energy consumption assuming a predetermined cluster size. Lazos et al. [57] considered a hierarchical key management structure for energy-aware secure multicast group communication in MANETs based on geographic routing. Their work assumed a fixed cluster size without identifying the optimal cluster size. Further, events that could happen in secure group communications including group join/leave and group partition/merge were not considered. Unlike previous work, [5, 6, 61] identified the optimal cluster size or its range in hierarchical clustering key management that would minimize the cost of clustering and rekeying. In particular, [5] proposed a hierarchical group key management protocol under which nodes are grouped into multiple clusters, and keys are distributed such that the intra-cluster communication is secured based on symmetric cryptography (i.e., using GDH.2) and the inter-cluster communication is secured based on asymmetric cryptography. They identified an optimal cluster size to tradeoff low communication cost for symmetric systems within a cluster versus high computation cost for asymmetric systems among cluster heads. However, no analysis was performed on the effect of group partition/merge and group leave/join events that can possibly occur in secure GCSs for MANETs.

We assume that nodes are equipped with GPS and therefore nodes are capable of knowing its geographical location. Instead of executing a clustering algorithm which wastes energy, nodes self-organize and group themselves into region-based subgroups. The region-based group key management protocol developed in this dissertation for MANETs derives from IGKMP [114] for decentralized group key management for scalability and efficiency. Unlike IGKMP, however, we adopt distributed key management within each region for robustness such that there is no single node acting as a key controller. All members within a region participate in key generation and management by following a distributed group key management protocol, thereby removing a single point of failure. Our hybrid decentralized-distributed region-based group key management protocol encompasses efficiency, scalability, and robustness without relying on central entities for key management. All nodes are homogeneous and no super-nodes are required. A major contribution of our proposed region-based group key management protocol is that our protocol is QoS-aware and we allow the optimal regional area size to be decided to minimize the network traffic due to group key management activities while satisfying security requirements, in response to group join/leave, mobility, and group partition/merge events.

A key design concept of our region-based group key management protocol is that a distributed protocol should be used for key generation and management within a region to achieve robustness. In the literature, there have been quite a few distributed group key management protocols proposed in recent years. *Group Diffie-Hellman* (GDH) [92] extends from the well-known two-party *Diffie-Hellman* (DH) key exchange protocol to allow a number of nodes to agree on a shared secret key without having a secure channel. In particular, GDH.2, GDH.3 in [91] and *Cliques* in [84] improved upon GDH. Based on GDH, Amir et al. [1, 2, 3] discussed a robust *contributory key agreement* (CKA) protocol resilient to group membership changes. Becker and Wille [9] proposed *Octopus* based on DH key exchange. In addition, ING

[46] extends from DH key exchange; BD [13] aims to reduce communication overheads based on public keys. Very recently distributed key management protocols based on logical key trees, e.g., *logical key hierarchy* (LKH) [74], have been proposed. Kim et al. [54, 55] proposed tree-based group key management protocols, STR and TGDH, combining the benefits of GDH and LKH. Rodeh et al. [83] suggested DLKH for which no key controller is needed and the logical key hierarchy is constructed by group members. DOFT, an extension of OFT [66], has been proposed in [35] to allow a group member to initiate access control and key generation. Wang et al. [103] extended OFT to be resistant to collusion attacks and Xu et al. [108] used OFT in batch rekeying protocols to improve scalability and efficiency. Lee et al. [58] developed distributed collaborated group key agreement protocols based on periodic batch rekeying [60] for performance optimization of distributed secure GCSs.

All the above distributed key management protocols incur high communication overheads as they have been designed to apply to the whole group. The region-based group key management protocol proposed in this dissertation allows any of these existing distributed key management protocols to apply at the subgroup level (i.e., at the intra-regional level) to achieve robustness without sacrificing efficiency. A key design in our region-based group key management protocol is to identify the “optimal regional area size” to minimize network traffic due to key management operations and mobility-induced events in MANETs. The optimal regional area size depends on the distributed key management protocol adopted. Our protocol is generic in general allowing any CKA protocol to be used. In this dissertation work, we exemplify how the optimal regional area size can be determined with GDH.

In our dissertation research, we integrate QoS-aware IDS protocols with region-based group key management to deal with both insider and outsider attacks for GCSs in MANETs. Our work is novel in that no prior work has been done to incorporate techniques to deal with both insider and outsider attacks for GCSs in MANETs. We demonstrate the security and performance benefits of the integrated scheme by performing comparative analyses against baseline secure group communication protocols based on individual rekeying, no intrusion detection, and/or no-region designs.

Chapter 3 SYSTEM MODEL

This dissertation concerns the design and analysis of a class of QoS-aware security protocols for secure group communications in wireless networks in terms of key management (i.e., threshold-based periodic batch rekeying and region-based group key management) and intrusion detection. The term "design and analysis" refers to design principles being used for building QoS-aware network protocols followed by analysis methodologies being applied for assessing system dependability properties so as to ascertain the benefit of network protocols designed. The term "QoS-aware" refers to the property that network protocols developed are conscious of QoS and are designed to satisfy security and performance QoS requirements dynamically by adjusting optimal operational settings identified in response to environmental changes at runtime. Our proposed QoS-aware security protocols can be generally applied to wireless networks with infrastructure support (e.g., a centralized key server exists) or MANETs without infrastructure support. The notion of a group is defined based on "connectivity." That is, group members must maintain connectivity for them to be in the same group. It is different from "mission-oriented" in the sense that a group may be partitioned into several groups due to network partition and node mobility. However, these partitioned groups will still continue with the same mission assigned throughout their lifetime. Later, when two or more partitioned groups merge into one, the merged group will still continue with the same mission execution. Therefore, mission execution is an application-level goal built on top of connectivity-oriented group communications.

This chapter is organized as follows. In Section 3.1, we describe security requirements for secure group communications in wireless networks. In Section 3.2, we describe two group key management protocols used for rekeying with full details. Section 3.3 discusses the underlying assumptions and factors that would affect the design of our proposed distributed intrusion detection protocol. In Section 3.4, we define the *attack model* including outsider and insider attacks in mobile GCSs. Finally, in Section 3.5, we describe our *security model* describing prevention/detection techniques used to deal with insider and outsider attacks considered.

3.1 Security Requirements for Secure Group Communication Systems

To secure wireless networks with or without infrastructure (e.g., where a centralized key server exists or there is no centralized entity such as MANETs), the following security attributes are considered in general: availability, confidentiality, integrity, authentication, and non-repudiation [51, 115, 116].

Availability guarantees the survivability of network services even under denial-of-service (DoS) attacks. This research does not consider DoS attacks but considers insider attacks by compromised entities in the system that may collude to make the system unavailable.

Confidentiality ensures that certain information is never disclosed to unauthorized parties. Confidentiality is required for network transmission of sensitive information, such as strategic or tactical military information.

Integrity makes sure that a message being transferred is never corrupted. Since there would be the loss of integrity by radio propagation impairment or malicious attacks on the wireless network, a message could be corrupted.

Authentication makes a node possible to ensure the identity of the peer node that it is communicating with. If there is no authentication, an adversary could disguise itself with identities of other nodes, thus obtaining unauthorized access to resource and sensitive information and interfering with the operation of other nodes.

Non-repudiation guarantees that the origin of a message cannot deny having sent the message. Non-repudiation is beneficial for detection and isolation of compromised nodes. When a node *A* receives a flawed message from a node *B*, non-repudiation permits *A* to accuse *B* utilizing this message and to convince other nodes that *B* is compromised. This dissertation research does not consider service of non-repudiation, as secure GCSs normally do not require non-repudiation.

For secure GCSs in MANETs, usually the first four features are recommended to be maintained, which are *availability*, *confidentiality*, *integrity*, and *authentication*. Non-repudiation is not required because most GCSs only support the publish/subscribe communication model which does not require non-repudiation. In addition to these security requirements mentioned above, as one of confidentiality properties, *secrecy* is generally considered in a secure GCS. Two unique *secrecy* properties are usually required for secure GCSs in dynamic networks that perform group activities such as group join or leave [83, 100]. First, ***forward secrecy*** assures that an adversary that knows a contiguous subset of old group keys cannot identify subsequent group keys. This guarantees that a member cannot know future group keys after it leaves the group. Second, ***backward secrecy*** ensures that an adversary that knows a subset of group keys cannot discover previous group keys. This guarantees that a new member that joins the group cannot learn any previous group keys. To maintain both backward and forward secrecy, rekeying (i.e., change the group key) should be performed when the group membership changes [60, 107].

3.2 Group Key Management Protocols

We describe two group key management protocols used in this dissertation research. First, when an infrastructure exists, we use a centralized key server-based management protocol called *LKH* for key management. Second, when no infrastructure exists, we employ *CKA* protocols to achieve a fully distributed group key management. In particular, we use the well-known GDH protocol.

3.2.1 Centralized Key Management Protocol

In a wireless environment with infrastructure support, a centralized key server can be used for key management. A new member (so called a receiver) contacts the key server to request the group key. The key server authenticates the receiver with a standard authentication protocol and establishes a secure channel that provides confidentiality, integrity, and authenticity. If the receiver is authorized, the key server sends group key information to the receiver. A group member encrypts messages with the group key to accomplish confidentiality and limit access to authorized receivers.

Without loss of generality, we consider that the centralized key server maintains a key tree based on *LKH* [74] to efficiently update the group key after a join or leave event to satisfy forward and backward secrecy requirements. Each node in the tree indicates a cryptographic symmetric key. The centralized key distribution center connects each group member with one leaf node of the tree and the following invariant will be always maintained; each group member knows all the keys from its leaf node up to the root node, but no other key in the key tree.

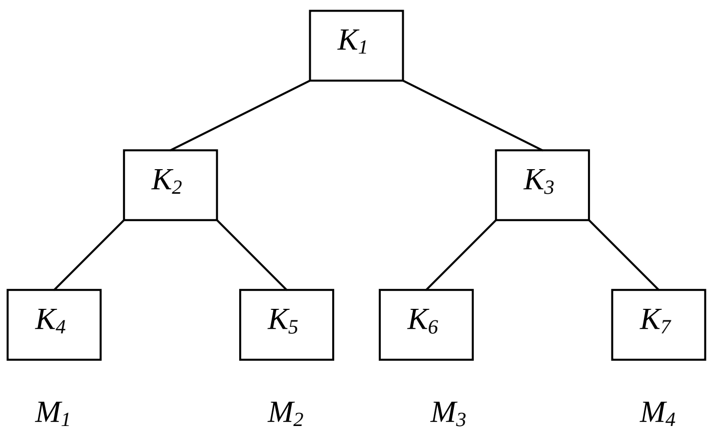


Figure 3- 1: An Example Hierarchical Key Tree.

We call the set of keys that a member knows the *key path*. Since all members know the key at the root node, that key plays a role as the group key. For instance, the key path for member M_2 in Figure 3-1 is composed of the keys K_5, K_2, and K_1. When a new member joins the group, the key server sends it all the keys on the key path over a secure channel. When a member leaves the group, the key server needs to update all the keys that the member knows, that is, all the keys on the key path. The main reason for utilizing such a key tree is to efficiently update the group key when join and leave events occur.

Note that the key update after a member leave event only requires a message of length $2k\log_2(N)$ bits, where k is the length of a key, and N is the number of members. Also, a key update operation after a new member join event requires a message of length $k\,(2\log_2 N - 1)$. One main benefit of using *LKH* is that each secure key update only requires a multicast message size that is logarithmic in the number of group members [57, 74] where a key server exists.

3.2.2 Distributed Key Management Protocol

In wireless environments with no infrastructure support, a CKA protocol would be used to achieve a fully distributed key management with no centralized key server. Our QoS-aware key management and IDS protocols developed in the dissertation research can allow any CKA protocol. Without loss of generality, we exemplify with a popular distributed key management protocol, GDH [92], as the key agreement protocol for secret key generation. In particular, we adopt GDH.3 in [92] since GDH.3 allows the use of fixed-sized messages and only a constant (and smaller) number of exponentiation operations executed by each participant. With these features, GDH.3 has been proposed for mobile devices with low computational capabilities [91].

Figure 3-2 summarizes the number of bits required in each stage of GDH.3 where N is the number of participants and v is the size of each intermediate value. For example, we apply GDH.3 as the key agreement protocol at both the intra-regional and inter-regional levels for the region-based group key management. Further, GDH.3 is also used in our proposed intrusion detection protocols in respect to rekeying operations due to its fully distributed feature. In [92], the secrecy property of GDH has already been proven.

$$Stage\ 1: upflow\ \ M_1 \rightarrow M_2 \rightarrow \cdots \rightarrow M_{N-2} \rightarrow M_{N-1}$$

$$message\ size\quad v\qquad v \ldots \ldots \ldots v \qquad\qquad = v(N-2)$$

$$Stage\ 2: broadcast\ \ M_{N-1} \rightarrow M_i\ \ where\ i \neq N-1$$

$$message\ size\qquad\qquad v \qquad\qquad = v$$

$$Stage\ 3: response\ \ M_i\ \ \ where\ i \neq N \rightarrow M_N$$

$$message\ size\qquad v\ from\ each\ \ M_i \qquad\qquad = v(N-1)$$

$$Stage\ 4: broadcast\ \ M_N \rightarrow M_i\ where\ i \neq N$$

$$message\ size\quad v(N-1)\ intermediate\ values\quad = v(N-1)$$

$$Total\ communication\ cost = 3v(N-1)$$

Figure 3- 2: Message Size Requirement in Each Stage of GDH.3.

GDH comprises four stages [92]. Each participant M_i shares a common base α and keeps its secret share N_i. The first stage collects contributions from all group members, M_1, M_2…, M_n. Specifically, M_1 raises α to the power of N_1, performing one exponential computation to generate α^{N1}, M_2 computes $\alpha^{N1\,N2}$ by raising α^{N1} to the power of N_2, and so on until M_{n-1} computes the last value $\alpha^{N1 \ldots\ Nn-1}$. After processing the up-flow message, M_{n-1} obtains $\alpha^{\prod\{N_k | k \epsilon [1,n-1]\}}$ and broadcasts this value in the second stage to all other participants. In the third stage, every M_i factors out its own exponent and forwards the result to M_n. In the final stage, M_n collects all inputs from all other participants, raises every one of them to the power of N_n and broadcasts the resulting n-1 values to the rest of the group. Every M_i receives this message in the form of $\alpha^{\prod\{N_k | k \epsilon [1,n-1] \cap k \neq i\}}$ and can easily generate the intended secret key K_n.

3.3 Intrusion Detection

We assume that nodes may be compromised with a variable rate. Recognizing principles in [40] that the behaviors of attackers are not random, we define three attacker functions to model this variable rate, namely, *logarithmic time attacker*, *linear time attacker*, and *polynomial time attacker*, as follows:

• ***Logarithmic time attacker***: The attacker increasingly takes longer time to compromise nodes in the system, following a logarithmic function curve. This models the scenario where the system has detected attackers (i.e., compromised nodes) and enhanced the defenses of the remaining nodes, making it increasingly harder for the attacker to compromise more nodes.

•***Linear time attacker***: The attacker compromises nodes one after the other with the node compromising rate linear to the number of compromised nodes in the system. This applies to the case in which compromised nodes do not collude and just perform constant time attacks.

•***Polynomial time attacker***: The attacker increasingly takes shorter time to compromise nodes in the system, following an exponential function curve. This models the scenario where the attacker learns secret information from compromised nodes in the system and exploits it to more easily compromise other nodes within a shorter time.

We assume that IDS will perform its function periodically. In our QoS-aware IDS protocols, the detection interval is dynamically adjusted in response to the accumulated number of intrusions that have been detected in the system.

Similar to the attacker behavior models above, we define three detection functions to model IDS activities, namely, *logarithmic periodic detection*, *linear periodic detection*, and *polynomial periodic detection*, as follows:

• ***Logarithmic periodic detection***: In this detection scheme, the system performs intrusion detection in a conservative way with a rate logarithmic to the number of compromised nodes that have been identified. This detection approach is usually applicable in a low or moderate level of hostile network environments. Further, this can be effective to save energy consumption introduced by IDS as well as to reduce false positives.

• ***Linear periodic detection***: This system performs IDS with a linear rate to the number of intrusions that have been detected in the system. Since this approach performs IDS more frequently than logarithmic periodic detection, it is a suitable detection approach when the employed IDS technique has high accuracy with relatively low number of false positives.

• ***Polynomial periodic detection***: This detection scheme aggressively performs IDS by increasing the detection rate in a polynomial fashion to the number of observed compromised nodes in the system. This scheme is effective for very high quality IDS with very low false negatives and false positives. Otherwise, using this scheme has the adverse effect of having more false positives and false negatives by performing IDS more than it needs.

To realize a fully distributed intrusion detection mechanism in MANETs, a host-based IDS protocol is preinstalled in each node. That is, each node has its own IDS to perform intrusion detection operations. Host-based IDS is characterized by two parameters, *p1* and *p2*, corresponding to the false negative probability and false positive probability, respectively. The system could perform host-based IDS to evict suspicious nodes. To alleviate collusion, the system performs voting-based IDS by which a majority of vote-participants must agree to evict a target node before the target node is evicted. The number of vote-participants (denoted by *m*) is a system parameter whose effect will be analyzed in this dissertation. Voting-based IDS is characterized by the false negative probability (P_{fn}) and false positive probability (P_{fp}) which depend on *p1* and *p2*, respectively, and the number of compromised nodes in the system.

We consider the *system lifetime* as the security-violation failure time of the mission-oriented GCS consisting of mobile groups. We consider two separate system failure definitions below for which we will analyze their effect on system lifetime:

- ***System Failure Definition 1 (SF1)***: the GCS fails when any group fails.
- ***System Failure Definition 2 (SF2)***: the GCS fails when all mobile groups fail.

These first system failure definition (SF1) applies to the case in which a security failure of any mobile group risks the entire system and causes the system to fail. For example, if the mission is to rescue military personnel by mobile groups then any compromised mobile group will cause the entire rescue operation to fail. The second system failure definition (SF2) applies to the case in which as long as there is one mobile group available in the GCS, the mission continues. An example is to reach a certain destination for tactical operations by mobile groups. In this case, any mobile group can operate independently of other mobile groups and the system fails only when all mobile groups fail. We will evaluate the effect of these two system failure definitions on the *MTTSF* of the system in Chapter 5.

A group enters a *security failure* state when one of the two conditions stated below is true:

- **Condition C1**: a compromised but undetected member requests and subsequently obtains data using the group key. The system is in a failure state because data have been leaked out to a compromised node, leading the *loss of integrity* [51] in a security sense.
- **Condition C2**: more than 1/3 of member nodes are compromised but undetected by IDS. We assume the *Byzantine Failure* model [40] such that when more than 1/3 of member nodes are compromised, the group is compromised, resulting in the *loss of availability* [51].

Note that under SF1, a group failure leads to the GCS system failure. Unless explicitly mentioned, the system failure definition will be based on SF1 in the dissertation research. If a member node is detected as compromised by IDS, depending on the rekeying policy used, the level of forward/backward secrecy will be determined. For example, when individual rekeying is used, the system won't allow the alleged member node to request data anymore and will evict the member immediately to satisfy the forward/backward secrecy requirement. When batch rekeying is used, however, since an alleged compromised node still possesses the group key, it may perform impersonation attacks to request for data using the group key, thus possibly leading to illegal data leak-out failures. After a node is detected as compromised and evicted from the system, it cannot rejoin the group again. That is, no recovery mechanism is available in the system to repair a compromised member and make it a trusted member node again.

3.4 Attack Model

Potential attacks in wireless networks can be classified based on criteria used. First, attacks in wireless networks can be divided into two categories: *passive attacks* or *active attacks* [51, 116]. Second, some researchers classify attacks based on the loss of security requirements: *loss of confidentiality*, *loss of integrity*, and *loss of availability* [51]. Lastly, attacks in wireless networks can be performed by two potential parties: *insider attacks* or *outsider attacks* [51, 114].

For the first classification (e.g., passive attacks versus active attacks), these two classes are also subdivided into several types of attacks [51].

Passive attack: When an unauthorized party gains access to an asset but does not modify its content, it is called a *passive attack*. Passive attacks can be either eavesdropping or traffic analysis (e.g., traffic flow analysis). *Eavesdropping* indicates that the attacker monitors transmissions of message content. *Traffic analysis* refers to analyzing patterns of data transmission for communications. That is, in a more subtle way, the attacker gains intelligence by monitoring transmitted data content [51].

Active attack: When an unauthorized party modifies a message, data stream, or file, it is called an *active attack*. Active attacks usually take the form of one of the following four types or combined form of them: masquerading, replay, message modification, and denial-of-service (*DoS*). *Masquerading* refers to the situation that the attacker impersonates an authorized party, thus gaining some authorized privileges. *Replay* means that the attacker monitors transmissions (e.g., passive attack) and retransmits messages as the legitimate user. *Message modification* attack occurs when the attacker modifies a legitimate message by deleting, adding to, changing, or reordering it. *Denial-of-Service (DoS)* attack takes place when the attacker blocks the normal use or management of communications facilities, for example, by causing excessive consumption of resources in networks [51].

For the second classification (e.g., loss of security requirements), Karygiannis and Owens suggest another class of categorization in security breach in terms of the loss of security requirements such as *confidentiality*, *data integrity*, and *network availability* [51]. Potential attacks in wireless networks can be categorized in terms of failing those security requirements.

Loss of confidentiality: Due to the broadcast and radio nature of wireless technology, *confidentiality* is a more challenging task to meet in wireless networks. This security risk includes attacks caused by passive eavesdropping, traffic analysis, masquerading, and message modifications [51].

Loss of integrity: *Data integrity* in wireless networks is an important issue as in wired networks. Since organizations frequently implement wireless and wired communications without appropriate cryptographic protection of data, data integrity may easily fail. Message modifications by hackers are a widely known attack to fail data integrity [51].

Loss of network availability: *Network availability* can be denied by some types of *DoS* attack. *DoS* attacks usually involve unnecessary excessive consumption of network resources so that legitimate users cannot continue their normal communications [51].

Lastly, for the third classification (e.g., insider attacks versus outsider attacks), based on legitimacy of an entity in a network, attacks can be classified into two classes: *insider attacks* versus *outsider attacks* [84]. If an entity is authorized to access system resources but employs them in a malicious way (i.e., not approved by those who granted the authorization), it is classified as an *insider attack.* On the other hand, an *outsider attack* is initiated from unauthorized or illegitimate user from the system [85]. More specifically:

Inside attackers exploit bugs in privileged system programs or poorly configured privileges, and then they may install backdoors or Trojan horse to facilitate subsequent acquisition of privileged access [85].

Outside attackers usually acquire access to an authorized account and tries to perpetrate an inside attacker. Both attackers may spoof network protocols to effectively acquire access to an authorized account [85].

This dissertation research deals with both outside and inside attackers. In general, an ***outside attacker*** would attempt to access an authorized account and then perpetrate as an inside attacker. Below are possible outsider attack scenarios [47]:

- An outside attacker can gain unauthorized access to a legitimate account by eavesdropping data packets or any message containing a secret key for more sophisticated attacks.
- An outside attacker can attempt to modify a data packet to break data integrity.
- An outside attacker may impersonate a group member to join a group.
- An outside attacker may forge packets.

Inside attackers, on the other hand, are due to compromised nodes disguised as legitimate members to disrupt the system. The following insider attack scenarios are considered in this dissertation research, as discussed in [47]:

- An adversary can snoop on the wireless channel to learn of secret information. For example, an adversary can eavesdrop messages sent by vote-participants against a target node, and can try to reconstruct the secret representing the final vote result against the target node.
- An adversary can collude with other compromised nodes so as to more efficiently compromise another node. For example, in the process of voting-based IDS (discussed

details in Chapter 5), an adversary can cast a negative vote against a trusted healthy node or cast a positive vote for a compromised node.

- An adversary can attempt to obtain secret information by communicating with other group members with its legitimate group key. When this happens, group information is leaked out and security failure, Condition C1 (see Section 3.3), has occurred.
- An adversary can leak the legitimately authorized secret information out to outside attackers. Further, an adversary can share their information with other nodes including both outside attackers and inside attackers to more easily compromise other nodes.

In this dissertation research, we develop threshold-based periodic batch rekeying protocols (Chapter 4) to deal with outsider attacks to preserve secrecy, confidentiality and data integrity. We also develop region-based group key management protocols (Chapter 7) for scalability to deal with outsider attacks in MANETs to preserve the same security properties. We develop QoS-aware intrusion detection protocols for mobile GCSs (Chapter 5) to deal with insider attacks which cannot be prevented using prevention techniques such as key-based encryptions or authentication. We integrate threshold-based periodic batch rekeying with QoS-aware IDS (Chapter 6) to deal with both insider and outsider attacks. Finally, we integrate region-based key management with QoS-aware IDS (Chapter 8) also to deal with both insider and outsider attacks and to achieve scalability and dynamic reconfigurability.

Section 3.5 below explains how outsider attacks are prevented or how insider attacks are tolerated in our security model.

3.5 Security Model

The class of QoS-aware key management and intrusion detection protocols developed in this dissertation research concerns both performance and security properties of the system. For security requirements, they deal with *secrecy*, *availability*, *confidentiality, integrity* and *authentication* requirements for secure group communications in the presence of malicious outsider/insider attacks.

Forward and ***backward secrecy*** [54] properties are preserved by means of rekeying to change the group key whenever there is a membership change. Our threshold-based periodic batch rekeying protocols in Chapters 4 and 6 encompass the special case of *individual rekeying* for which there is no secrecy violation, as well as the general case of batch rekeying for which the secrecy requirement, particularly *forward secrecy*, can be traded off for better service response time. Note that forward and backward secrecy falls under confidentiality.

Service availability is achieved by maximizing *MTTSF* of the system in the presence of *insider attacks* in our proposed distributed intrusion detection protocols (Chapters 5, 6, and 8). For example, introducing adaptive IDS that dynamically adjusts its intrusion detection interval based on the accumulated number of intrusions will maximize *MTTSF* of the system by removing compromised nodes adaptively.

For ***authentication***, we assume that each member has a private key and its certified public key available for *authentication* purposes. When a new member joins a group, the new member's identity is authenticated based on the member public/private key pair by applying the challenge/response mechanism. For example, in our proposed region-based group key management protocol, source authentication is ensured during regional, leader and group key generation. That is, when a regional key or a leader key is generated through *CKA*, the source authentication of a participating member is achieved by using the private/public key pair. In summary, preloaded private/public key pairs are used for source authenticity during rekeying operations and voting-based intrusion detection procedures.

Confidentiality is ensured by using the secret key shared by involved parties to encrypt information exchanged among the parties. Specifically, for instance, in our region-based group key management protocol, a regional key is only used by members within a region; a leader key is used among leaders; and a group key is only used by group members for group communication activities. When a group key is generated from the leader key, i.e., $K_G = \text{MAC}(K_{RL}, c)$, based on MAC (Message Authentication Code) using the current leader key K_{RL} and a fresh counter c, the group key is encrypted by the regional key K_R for a leader to disseminate to its members in the region. Thus, confidentiality is maintained because only members in the region have and can use the shared regional key to decrypt the group key.

Finally, to preserve ***integrity***, a member can generate a message authentication code (MAC) using the secret key it shares with other group members as the MAC key when a message is sent. It is computationally impossible to alter the content of the message without being detected by a receiver that shares the secret key. For example, in our region-based group key management protocol, this can be applied for intra-regional communication between a leader and its regional members using the shared regional key, for inter-regional communication among leaders using the shared leader key, or among members using the shared group key. Further, in our proposed voting-based IDS each vote from a vote-participant is disseminated with a MAC, e.g., MAC (K_G, S) where S refers to a secret share and K_G is a group key. Thus, it is impossible for an outside attacker to modify the message without knowing the secret key, K_G, which is only possessed by legitimate members. Replay attack can be prevented by incorporating a sequence number into each packet.

Chapter 4 THRESHOLD-BASED PERIODIC BATCH REKEYING

In this chapter, we present our results on the design and analysis of a class of threshold-based periodic batch rekeying protocols for achieving QoS-aware key management in wireless networks with a centralized key server to deal with outsider attacks. The content is largely based on our work published in [20, 21]. We also report our experience of applying the design and optimization principles to wireless networks without infrastructure support, e.g., MANETs.

The organization of this chapter is as follows. Section 4.1 provides the background and describes the designs of these threshold-based periodic batch rekeying protocols. In Section 4.2, we describe the behaviors of these key management protocols by means of mathematical models and identify a set of design parameters that can characterize the operational and workload conditions of the system. Section 4.3 reports numerical results generated from analytical models and demonstrates that there exists an optimal batch rekey interval that maximizes performance of the system while satisfying security requirements of the system. In Section 4.4, we summarize this chapter and report experiences learned when applying the design and optimization principles to MANET environments.

4.1 Threshold-based Periodic Batch Rekeying

For efficient key management to deal with outside attackers, we propose new threshold-based protocols for periodic batch rekeying and demonstrate that there exists an optimal batch rekey interval for each protocol to minimize the communication overhead per rekeying operation, when given a set of parameter values characterizing the environmental conditions, such as the arrival ratio of join or leave requests or the probability of trustworthiness in the network [75]. We compare these threshold-based periodic batch rekeying protocols under the same set of environmental conditions, and identify conditions under which a protocol would perform the best in terms of the minimum communication overhead per join/leave operation without violating constraints in secrecy (security) and rekeying delay (performance) requirements. To reveal the batch rekey interval that optimally explores the tradeoff between acceptable secrecy violation and rekeying delay in wireless networks, we develop probability models and Stochastic Petri net (*SPN*) models to evaluate our proposed protocols measured by the communication overhead per operation, probability of secrecy violation, rekeying delay, and batch rekey interval.

4.1.1 Background

We assume that periodic batch rekeying is employed in the resource-constrained wireless network to alleviate rekeying overheads in terms of the communication overhead incurred due to join or leave requests. We assume that a user cannot join the group unless it is authorized by the authentication server. In this case, the join request is identified as a "trusted" join. If a user can join without authentication, then we term that join as "untrusted" join. Untrusted joins are not allowed in our model. A leave request also may be "trusted" or "untrusted." A trusted leave is the one issued by a user that voluntarily leaves the group, for example, because it has moved to another location. On the other hand, a leave is untrusted if the leave is due to the eviction of the group member. If rekeying does not take place immediately after an untrusted leave, a period of security vulnerability occurs until rekeying takes place. When processing a leave operation, the key server is able to differentiate a trusted leave operation from an eviction operation. The probability of trustworthiness (P_t) for leave operations thus can be computed by the key server as the ratio of the number of trusted leave operations over the total number of leave and eviction operations statistically collected by the key server periodically. Figure 4-1 describes the system environment to evaluate our proposed protocols.

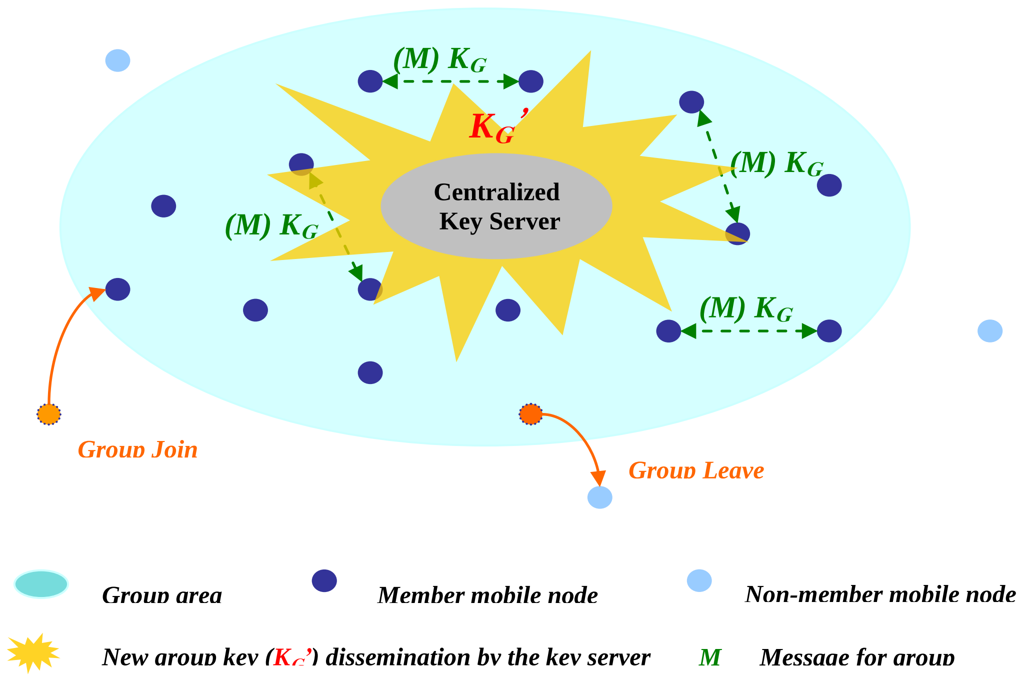


Figure 4- 1: Centralized Key Management for Wireless Secure Group Communications.

4.1.2 Protocol Description

Our batch rekeying protocols are designed based on the notion of *thresholds* to govern the maximum number of requests (either join or leave, or both) that can be accumulated in the key server, beyond which rekeying will be performed. As a baseline, we consider a periodic batch rekeying protocol (called *ULT* below) for which only one threshold, say *k3*, is used. When *k3* is reached, rekeying will be performed. We also consider more sophisticated periodic batch rekeying protocols for which two thresholds, say *k1* and *k2*, are used and when either *k1* or *k2* has reached, rekeying will be performed. By using thresholds, a threshold-based policy thus identifies the set of states in which rekeying will be performed, thereby implicitly determining the time interval between two rekeying operations. The special case in which individual rekeying, i.e., a rekeying operation is performed whenever there is a join/leave operation, can be obtained by setting the two thresholds to 1 in the latter two protocols.

The behaviors of threshold-based periodic batch rekeying protocols proposed can be described by a state machine with a 3-component state representation (*a*, *b*, *c*), where *a* is the number of trusted join requests, *b* is the number of trusted leave requests, and *c* is the number of untrusted leave requests, respectively. We consider three different threshold-based periodic batch rekeying protocols as follows:

- ***Untrusted Leave Threshold-based (ULT)***: This protocol has only one threshold, say *k3*, to check against the number of untrusted leave requests (i.e., *c* in the state representation). This protocol only guards against untrusted leave requests irrespective of the traffic pattern of trusted users. For the special case in which *k3* is 1, *ULT* degenerates to individual rekeying for untrusted requests. We use *ULT* as a baseline protocol against which we compare the performance characteristics of two other more sophisticated batch rekeying protocols described below.
- ***Trusted and Untrusted Double Threshold-based (TAUDT)***: This protocol has two thresholds, *k1* and *k2*, with *k1* checking against the number of trusted requests (i.e., $a+b$) and *k2* checking against the number of untrusted leave requests (i.e., *c*).
- ***Join and Leave Double Threshold-based (JALDT)***: This protocol has two thresholds, *k1* and *k2*, with *k1* checking against the number of trusted join requests (i.e., *a*) and *k2* checking against the number of leave requests including both trusted and untrusted requests (i.e., $b+c$).

To consider untrusted requests, the probability of trustworthiness (P_t), which indicates the percentage of trusted requests received, is given in all three protocols. For untrusted join requests, the key server does not accept the new node's join request through authentication and authorization. Thus, only untrusted leave requests need to be considered by the key server. Recall that this proposed protocols use a centralized key management protocol, *LKH*, for rekeying operation as described in Chapter 3. In Figure 3-1 (of Chapter 3), K_1, K_2, K_3, and K_4 refer to the group keys updated in each interval. Rekeying is performed only at the end of the batch rekey interval defined as the period between two successive group keys updates, such as between K_1 and K_2 labeled in Figure 3-1.

Two application-specific constraints are considered here: probability of secrecy violation (P_v) and delay (*D*) incurred due to periodic batch rekeying. The delay parameter (*D*) refers to the average latency experienced per join or leave operation. The probability of secrecy violation (P_v) is measured by the proportion of the time that the secrecy requirement is violated. Note that we only need to consider *forward secrecy* violation (caused by delayed rekeying for leave requests). That is, when a new member joins the group, there is no backward secrecy violation because no key is ever issued to the new member until the end of the batch rekey interval. On the other hand, when an untrusted member requests to leave the group, there is a forward secrecy violation since the untrusted member does not leave immediately right after it requests a leave operation, and has to stay until the end of batch rekey interval, allowing it to learn group information. As a result, by the probability of secrecy violation, we refer to the proportion of the time that the forward secrecy is violated due to the presence of untrusted users having requested to leave the group.

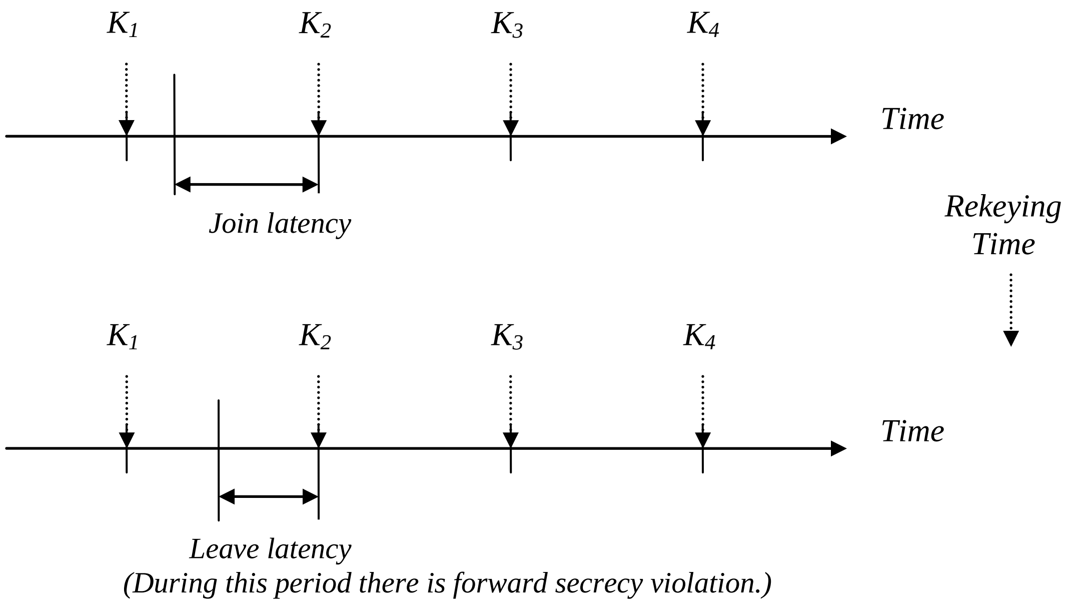


Figure 4- 2: Periodic Batch Rekeying with respect to Join and Leave Events.

Note that we do not distinguish join rekey interval from leave rekey interval because join and leave events are aggregated and processed at the end of each batch interval through rekeying. The optimal batch rekey interval (*T*) is the interval at which the overhead is minimized while satisfying the two application-level constraints in terms of the probability of secrecy violation (P_v) and delay (*D*) caused by the postponed rekeying, e.g., 5% of P_v and 5 *s* of *D*.

A simple optimization feature is used to reduce communication overhead taking advantage that the key server in our design has both join and leave requests for rekeying. That is, a new join member can take the place of a leave member in the key tree. Thus, for each pair of join and leave requests, the key server only needs to generate new keys along the paths of the leave members and give the new keys to the new join members. Recall that a state in our design is represented by (*a*, *b*, *c*) where *a* is the number of trusted join requests, *b* is the number of trusted leave requests, and *c* is the number of untrusted leave requests. The key server applies the following procedures when performing a rekeying operation at the end of each batch interval:

- if $a > b+c$, then the server will process $b+c$ join-leave request pairs before processing $a - (b+c)$ join requests;
- if $a = b+c$, then the server will process $b+c$ join-leave request pairs;
- if $a < b+c$, then the server will process a join-leave request pairs before processing $(b+c)-a$ leave requests.

4.2 Performance Analysis

Table 4-1 summarizes the notation used for parameters in the proposed threshold-based periodic batch rekeying protocols. We assume that the inter-arrival times of join requests and leave requests are exponentially distributed with rates λ and μ, respectively. This assumption allows us to construct an *SPN* model that can be evaluated using tools such as *SPNP v4* [30]. The assumption of exponential distribution can be relaxed easily by defining other time distributions and evaluating the model using *SPNP v6* [31].

4.2.1 Single Threshold-based Batch Rekeying Model

For *ULT*, we derive analytical closed-form solutions below to calculate the minimum communication overhead per operation (S), the probability of secrecy violation (P_v), and the delay (D) occurred due to periodic batch rekeying.

Let T be the average batch rekey interval in *ULT*, which can be calculated as follows:

$$T = \frac{k3}{\mu(1 - P_t)} \qquad \text{(4- 1)}$$

where $1/\mu(1\text{-}P_t)$ is the average inter-arrival time of an untrusted leave request.

Thus, for *ULT*, at the end of each batch rekey interval, the state of the system represented by (a, b, c) (see Table 4-1 for their definitions) will have the following state variable values:

$$a = \lambda \times P_t \times T, b = \mu \times P_t \times T,\ c = \mu \times (1 - P_t) \times T \qquad \text{(4- 2)}$$

Consequently, based on the procedure used by *ULT* for performing rekeying at the end of each batch interval, the total communication overhead bits (C_m) in *ULT* can be computed based on LKH for rekeying operations as follows:

$$if a \geq (b + c) \tag{4- 3}$$

$$then\, J \times (b + c) \times 2log_2 N + J \times (a - b - c) \times (2log_2 N - 1)$$

$$= J \times a \times 2log_2 N - J \times (a - b - c)$$

$$else\ if\ a < (b + c)$$

$$then\, J \times a \times 2log_2 N + J \times (b + c - a) \times 2log_2 N$$

$$= J \times (b + c) \times 2log_2 N$$

Table 4- 1: Notation.

Symbol	**Meaning**
$\boldsymbol{\lambda}$	Arrival rate of join requests
$\boldsymbol{\mu}$	Arrival rate of leave requests
$\boldsymbol{P_t}$	Probability of trustworthiness, i.e., probability that a user request is issued from a trusted user
$\boldsymbol{T_b}$	Average overhead (delay) for broadcasting in the wireless network (*s*)
BW	Network bandwidth (*Mbps*)
$\boldsymbol{C_m}$	Communication overhead bits in each rekeying state (*bits*)
$\boldsymbol{S_{cm}}$	Average communication overhead (delay) per rekeying operation (*s*)
S	Average communication overhead (delay) per join/leave operation (*s*)
$\boldsymbol{P_v}$	Average probability of secrecy violation
D	Average delay occurred per join/leave operation (*s*)
T	Average batch rekey interval (*s*)
N	Total number of members in the group
J	Length of each key (*bits*)
a	Number of trusted join requests
b	Number of trusted leave requests
c	Number of untrusted leave requests
ULT	Untrusted Leave Threshold-based: This protocol has only one threshold, *k3*, to check against the number of untrusted leave requests (i.e., *c* in the state representation)
TAUDT	Trusted and Untrusted Double Threshold-based: This protocol has two thresholds, *k1* and *k2*, with *k1* checking against the number of trusted requests (i.e., *a*+*b*) and *k2* checking against the number of untrusted leave requests (i.e., *c*).
JALDT	Join and Leave Double Threshold-based: This protocol has two thresholds, *k1* and *k2*, with *k1* checking against the number of trusted join requests (i.e., *a*) and *k2* checking against the number of leave requests (i.e., *b*+*c*)
k1	First threshold used by *TAUDT* and *JALDT*
k2	Second threshold used by *TAUDT* and *JALDT*
k3	Only threshold used by *ULT*

Let S_{cm} be the communication overhead (referring to the communication delay) required for performing batch rekeying with the time unit. Let T_b be the overhead for broadcasting in the wireless network. Then, S_{cm} can be calculated as the sum of T_b and the packet transmission time calculated as the communication overhead bits (C_m) given by Equation 4-3 divided by the wireless bandwidth, i.e.,

$$S_{cm} = T_b + \frac{C_m}{BW} \qquad (4\text{- }4)$$

Note that T_b includes the wireless channel contention time and the wireless propagation time for broadcasting a message, both of which can be monitored by the key server. In practice, the key server can timestamp every broadcast message prior to transmission, and, based on the timestamps of acknowledgements returned from members, deduce T_b as the average time difference minus the transmission time.

The average communication overhead per join/leave operation (S) in *ULT* for rekeying is simply equal to the total overhead divided by the number of leave/join operations, i.e.,

$$S = \frac{S_{cm}}{(a+b+c)} \qquad (4\text{- }5)$$

The probability of secrecy violation (P_v) due to periodic batch rekeying in *ULT* is calculated as the proportion of time in which forward secrecy is violated because of the presence of untrusted leave requests, i.e.,

$$P_v = \frac{\left({(k3-1)}/{k3}\right) \times T + S_{cm}}{(T + S_{cm})} \qquad (4\text{- }6)$$

Here $T+S_{cm}$ in the denominator is a base observation period and $\{(k3-1)/k3\} \times T + S_{cm}$ in the numerator is the duration within the base observation period in which forward secrecy is violated. Note that when $k3$=1, the probability of secrecy violation (P_v) is simply $S_{cm}/(T+S_{cm})$ because in this special case an untrusted leave request arrives at the system in every T time interval on average and as soon as it arrives, the system immediately takes S_{cm} to perform rekeying to process the arriving untrusted leave request (because $k3$ = 1), during which forward secrecy is violated.

The delay per join/leave operation (*D*) in *ULT* is obtained by:

$$D = S + \frac{T}{2} \qquad (4\text{- }7)$$

Here *T/2* is the average waiting time for batch rekeying as experienced by an operation and *S* is the average communication overhead per join/leave operation. Through a sanity check that compares *D* with the response time per operation, we validate that the calculated *D* is almost the same as the response time per operation, thus justifying its use. Here the response time, *R*, is obtained by using *Little's law* *R = n/X* [84], where *n* is the average number of requests and *X* is the throughput of the system.

4.2.2 Double Threshold-based Batch Rekeying Model

For *TAUDT* and *JALDT*, there are too many states to yield closed-form analytical expressions. Therefore, an *SPN* model is developed to measure performance metrics including P_v, *D*, *T*, and *S*. Figure 4-3 shows our *SPN* model. For convenience, Table 4-2 lists the places, transitions, transition rates, arcs and arc multiplicities used in the *SPN* model. We first briefly introduce the nomenclature necessary for the comprehension of an *SPN* model [31]. An *SPN* model consists of entities including transitions, places, arcs, and tokens. A token is used as a marker; it can be used to represent a user request. A place is a token place-holder to contain tokens representing join and leave requests; it is normally given a distinct name that conveys the meaning of a state component, e.g., place *a* in Figure 4-3 holds the number of trusted join requests, place *b* holds the number of trusted leave requests, and place *c* holds the number of untrusted leave requests (corresponding to *a*, *b* and *c* in Table 4-1). The function *mark* (*p*) is used to return the current number of tokens held in place *p*. Typically, state components in the state representation of the underlying *Markov* or *semi-Markov* model correspond to places in an *SPN*. Since a state in our model has three components, namely, *a*, *b*, and *c*, three places, namely, *a*, *b* and *c* are created for these state components, respectively. Place *tmp* is a temporary placeholder, which does not correspond to any state component and is used to hold newly arriving leave requests.

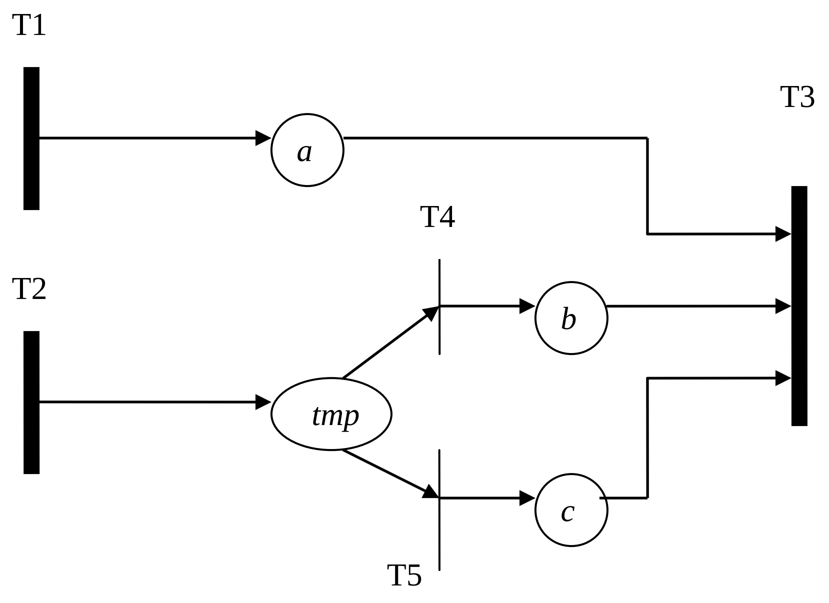


Figure 4- 3: SPN Model for TAUDT and JAUDT.

Table 4- 2: Places, Transitions, Transition Rates, Arcs and Arc Multiplicities for the *SPN* Model in Figure 4-3.

Place	**Meaning**		
a	*mark(a)* indicates "*a*" (number of trusted join requests).		
b	*mark(b)* indicates "*b*" (number of trusted leave requests).		
c	*mark(c)* indicates "*c*" (number of untrusted requests).		
tmp	*mark(tmp)* = 1 indicates that a leave request has just arrived; *mark(tmp)* is always 1 or 0.		
Transition	**Type**	**Rate or Probability**	
T1	Timed	λP_t	
T2	Timed	μ	
T3	Timed	$1/S_{cm}$	
T4	Immediate	*Pt*	
T5	Immediate	$1\text{-}P_t$	
Input arc	**Multiplicity**	**Output arc**	**Multiplicity**
tmp – T4	1	***T1– a***	1
tmp – T5	1	***T2– tmp***	1
a – T3	*mark(a)*	***T4– b***	1
b – T3	*mark(b)*	***T5– c***	1
c – T3	*mark(c)*		

A *transition* represents an event. If a *timed* transition is fired in an *SPN*, then it means that an event associated with the transition has occurred, e.g., a leave request arrives after a time exponentially distributed (or generally distributed) has elapsed in the *SPN* model. For modeling convenience, we also allow *immediate* transitions to exist in an *SPN* model. An immediate transition occurs instantaneously without taking any time when the transition fires.

Arcs connect places to transitions. We differentiate *input arcs* from *output arcs*. An *input arc* goes from an input place to a transition, while an *output arc* goes from a transition to an output place. An *arc* can be associated with a multiplicity to indicate the number of tokens associated with the arc; the default is 1 if not specified. A transition can be optionally associated with an *enabling function* to explicitly check conditions to be satisfied to allow the transition to be fired. An enabling function will return either *true* or *false* depending on the current state of the system. For example, *TAUDT* will perform rekeying when place *c* holds a number of tokens equal to the *k2* threshold, or places *a* and *b* altogether hold a number of tokens equal to the *k1* threshold. In Figure 4-3, a transition can fire if the following two conditions are satisfied: (a) there are at least *m* tokens in each of its input places connected to it by an input *arc* with multiplicity of *m*; (b) the associated enabling function (if one is assigned) returns *true*.

Below we explain how the *SPN* model shown in Figure 4-3 is constructed:

- When a trusted join request arrives, a token is created to move to place *a* used to hold the number of trusted join requests. This is modeled by transition *T1* with rate λP_t. Note that untrusted join requests will be detected by the key server, so the transition rate here is λP_t only to account for the arrival rate of trusted join requests. We use a parameter P_t to denote the probability of trustworthiness, that is, the probability that a request is issued from a trusted entity.
- When a trusted or untrusted leave request arrives, a token is created to move to *tmp*. This is modeled by transition *T2* with rate μ. Our model distinguishes trusted requests from untrusted requests. If the leave request is from a trusted entity, the token in *tmp* flows to *b*; otherwise, the leave request is untrusted and the token in *tmp* flows to *c*. Immediate transition *T4* is associated with probability P_t, while transition *T5* is associated with probability 1-P_t. They are fired as soon as its input place, e.g., *tmp*, contains a token, after which the token will be moved from *tmp* immediately to *b* with probability P_t, and to *c* with probability 1-P_t.
- Under *TAUDT* or *JALDT*, when a rekeying condition is satisfied, i.e., when either the *k1* or *k2* threshold is reached, rekeying is performed. This is modeled by associating an enabling function with transition *T3* specifying the rekeying condition to be satisfied and firing *T3*

when it is so. Based on the threshold control policies, the enabling function of *T3* for *TAUDT* is *if mark(a) + mark(b) = k1 or mark(c) = k2, then return true; otherwise return false.* The enabling function of *T3* for *JALDT* is *if mark(a) = k1 or mark(b)+ mark(c) = k2, then return true; otherwise return false.* After a rekeying operation is processed by the key server, all the tokens in *a*, *b*, and *c* (representing the join/leave operations accumulated at the server over the batch interval period) are removed through transition *T3* and the state system goes back to the initial state (0, 0, 0), i.e., *mark*(*a*) = 0, *mark*(*b*) = 0 and *mark*(*c*) = 0.

Table 4-3 lists the enabling functions associated with transitions in the *SPN* model for the *TAUDT* and *JALDT* protocols, reflecting their respective control behaviors for firing the transitions. The average communication overhead per operation (*S*) is obtained by assigning a reward of S_{cm}/(*mark(a)+mark(b)+mark(c)*) to each rekeying state in which the enabling function of *T3* returns *true*, where S_{cm} is calculated by Equation 4-4 whose value depends on the values of *a*, *b*, and *c* in each rekeying state. Specifically, the following formula is used to calculate *S* in the *SPN* model:

$$S = \sum_{i \in R}^{all} P(i) \times \frac{S_{cm}}{(mark(a) + mark(b) + mark(c))} \qquad \text{(4- 8)}$$

Here *R* denotes the set of rekeying states and *P(i)* denotes the steady-state probability of the system being in state *i*, which we could easily obtain by evaluating the *SPN* model using *SPNP* [28, 81]. Under *TAUDT* and *JALDT*, secrecy is violated when there is at least one untrusted leave request in the system. Thus, the probability of secrecy violation P_v is obtained by assigning a reward of 1 when *mark(c)* > 0, calculated as follows:

$$P_v = \sum_{i \in V}^{all} P(i) \times r_i \qquad \text{(4- 9)}$$

Here *v* denotes the set of states in which *mark (c)* > 0, r_i is 1, and *P(i)* is the probability that the system is in state *i* in the steady-state.

In order to obtain the average batch rekey interval *T* under *TAUDT* and *JALDT*, we transform the *SPN* model shown in Figure 4-3 into one in which all rekeying states become absorbing states. Then, in this transformed *SPN* model with absorbing states, by assigning a reward of 1 to all states except the absorbing states, *T* can be computed by the expected cumulative reward until absorption, *E[Y(∞)]*, since this *mean time to absorption* corresponds to the average time it takes to reach an absorption state in which rekeying will be performed. Specifically, in the transformed *SPN* model with rekeying states as the absorbing states, *T* is calculated as follows:

$$T = E[Y(\infty)] = \sum_{i \in S}^{\infty} r_i \int_0^{\infty} P_i(t) dt \qquad (4\text{- }10)$$

Here *S* denotes the set of all states except the absorbing states in the transformed *SPN* model, r_i=1, and $P_i(t)$ is the probability of state *i* at time *t.* An *SPN* evaluation tool such as *SPNP* [28, 81] can be readily applied to compute *T* based on Equation 4-10. Once *S* and *T* are obtained from Equations 4-8 and 4-10, the average delay per operation *D* can be calculated based on Equation 4-7 for *TAUDT* and *JALDT*.

4.3 Numerical Analysis and Comparison

This section presents and analyzes the numerical results obtained from applying the mathematical models developed for *ULT*, *TAUDT* and *JALDT*. In all cases presented, the number of members in the group (*N*) is set to 1024 (representing a large dynamic group), the length of each key (*J*) is 64 bits, T_b=5*ms*, and the bandwidth (*BW*) is 1 *Mbps*. Changing these parameter values will affect the scale of the results but does not affect the trend. On the other hand, we change the values of other key parameters including the ratio of the arrival rate of join requests to the arrival rate of leave requests (λ: μ) and the probability of trustworthiness (P_t) to see their effects on the results.

We organize the presentation as follows. First, we show that for each of the three threshold-based periodic batch rekeying protocols proposed (*ULT*, *TAUDT*, and *JALDT*) an optimal batch rekey interval (*T*) exists that would minimize the cost per join/leave operation (*S*) while satisfying the requirement constraints in terms of delay (*D*) and secrecy violation (P_v) in Sections 4.3.1, 4.3.2 and 4.3.3, respectively. Then in Section 4.3.4, we compare these threshold-based protocols head-to-head under identical system conditions characterized by the probability of trustworthiness (P_t) and the ratio of λ: μ, and identify the best protocol that minimizes *S* among all. We used a *log* scale (*base 10*) to represent the values measured.

Table 4- 3: Transitions and Associated Enabling Functions in the *SPN* Model.

Transition	Enabling function
T1	
TAUDT	*If mark (a)+mark(b) < k1 and mark(tmp) =0, then return true; otherwise return false.*
JALDT	*If mark (a)< k1 and mark(tmp) =0, then return true; otherwise return false.*
T2	
TAUDT	*If mark (c) < k2 and mark(tmp) =0, then return true; otherwise return false.*
JALDT	*If mark(b)+mark (c) < k2 and mark(tmp) =0, then return true; otherwise return false.*
T3	
TAUDT	*If mark(a)+mark(b)= k1 or mark(c) =k2, then return true; otherwise return false.*
JALDT	*If mark(a) = k1 or mark(b)+ mark(c) =k2, then return true; otherwise return false.*
T4	
TAUDT	*If mark (a)+mark(b) < k1 and mark(tmp) =1, then return true; otherwise, return false.*
JALDT	*If mark (a)< k1 and mark(tmp) =1, then return true; otherwise return false.*
T5	
TAUDT	*If mark (c) < k2 and mark(tmp) =1, then return true; otherwise return false.*
JALDT	*If mark(b)+mark (c) < k2 and mark(tmp) =1, then return true; otherwise return false*

4.3.1 Untrusted Leave Threshold-Based (ULT) Batch Rekeying

Recall that the *ULT* batch rekeying protocol is our baseline protocol which *TAUDT* and *JALDT* will be compared against. It only has one threshold, *k3*, to guard the number of untrusted leave requests (i.e., *c*). Figure 4-4 shows the effect of varying *k3* on the probability of secrecy violation, P_v, in *ULT* while setting $\lambda: \mu$ = 1: 0.5 and P_t = 0.9. Other $\lambda: \mu$ and P_t values exhibit similar trends and are not shown here. As we can see, P_v increases as *k3* increases. The reason is that *k3* checks against the number of untrusted leave requests (*c*) in the key server. Therefore, increasing *k3* means that there are more untrusted leave requests in the key server accumulated until rekeying is performed, thus resulting in a higher probability of secrecy violation. Note that when *k3* = 1, P_v is 0. That means that as soon as the key server accepts an untrusted leave request, it performs a rekeying operation immediately, in which case there is no secrecy violation and forward secrecy is preserved without any violation at the expense of performance degradation.

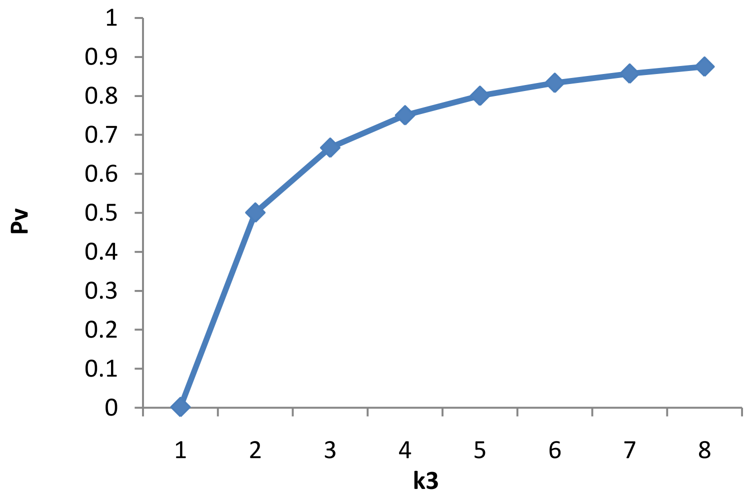


Figure 4- 4: *P_v* versus *k3* under the *ULT* Scheme.

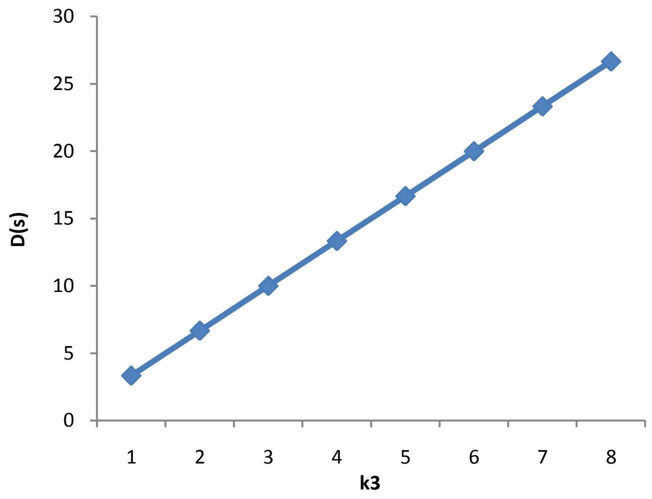


Figure 4- 5: *D* versus *k3* under the *ULT* Scheme.

Figure 4-5 shows the effect of changing *k3* on the delay (*D*) incurred due to periodic batch rekeying in *ULT*. As shown in Figure 4-5, *D* increases as *k3* grows. The reason is that when a higher threshold (*k3*) is applied for batch rekeying, it takes more time to accumulate the number of untrusted leave requests by the key server to reach the threshold, thus increasing *D*.

Figure 4-6 shows the average communication overhead per join/leave operation (*S*) as *k3* increases. As expected, *S* decreases as *k3* increases. The optimal *k3* value that minimizes *S* while satisfying the imposed constraints on *D* and P_v, however, is not infinity. For example, when $D = 5\ s$ and $P_v = 5\ \%$, *k3* is 1. The corresponding optimal batch rekey interval (*T*) that minimizes *S* while satisfying *D* and P_v in this case can be readily calculated as 6.67 *s* based on Equation 4-1.

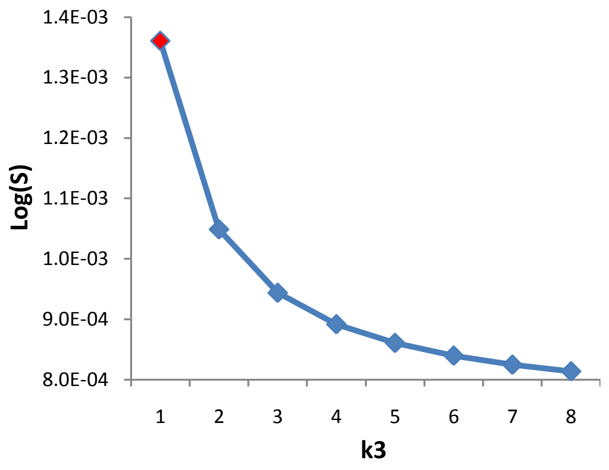


Figure 4- 6: *S* versus *k3* under the *ULT* Scheme.

4.3.2 Trusted and Untrusted Double Threshold-based (TAUDT) Batch Rekeying

Recall that *TAUDT* has two thresholds, *k1* and *k2*, with *k1* guarding against the number of trusted requests *(a+b)* and *k2* guarding against the number of untrusted requests *(c)*. Figure 4-7 shows the effect of *(k1*, *k2)* on P_v in *TAUDT* with $\lambda : \mu = 1: 0.5$ and $P_t = 0.9$. As *k1* increases, P_v increases (except when *k2* = 1 representing the special case that secrecy is perfect) because a higher threshold contributes to more states having violated the secrecy requirement. P_v also increases as *k2* increases in general until *k2* reaches the threshold (*k2* > 2) beyond which P_v is insensitive to the increase of *k2*.

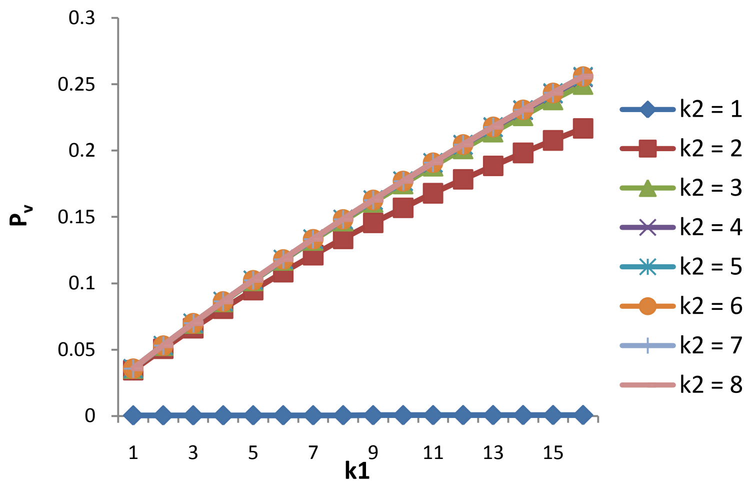


Figure 4- 7: P_v versus (*k1*, *k2*) under the *TAUDT* Scheme.

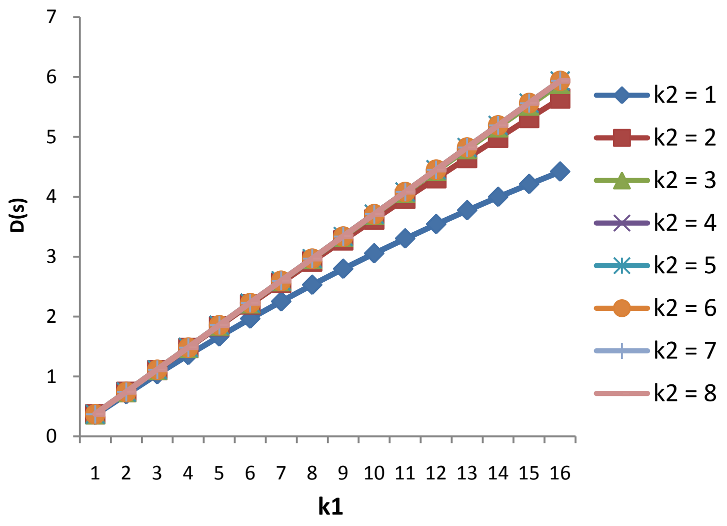


Figure 4- 8: *D* versus *(k1, k2)* under the *TAUDT* Scheme.

The reason is that with P_t = 0.9 most arrivals are trusted requests and thus the effect of *k1* as a threshold dominates the effect of *k2*. We observe that, nevertheless, when P_t decreases, P_v becomes more sensitive to *k2*, and the P_v versus *(k1*, *k2)* curves become more distinct for different *k2* values.

Figure 4-8 shows *D* versus *(k1*, *k2)*. We observe that *D* increases as *k1* increases, because using a higher threshold to aggregate more join or leave requests will result in a higher latency. Here, *D* also increases as *k2* increases although the effect of *k2* is not as significant as *k1* due to a high P_t used. Again, we observe a more significant effect of *k2* on *D* when we decrease P_t.

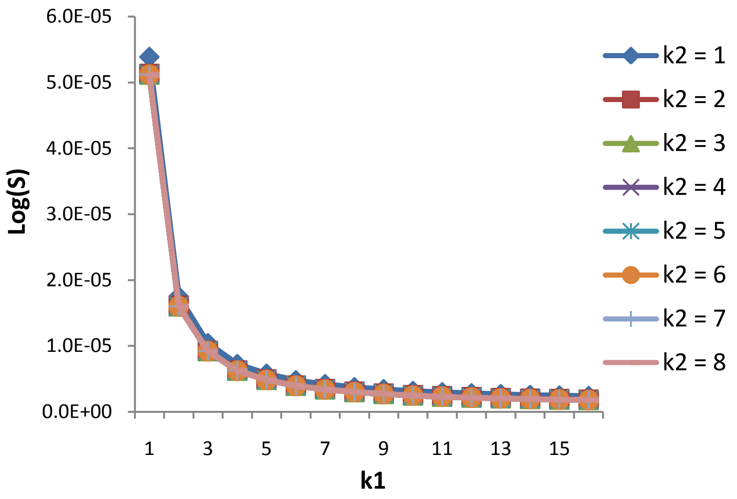


Figure 4- 9: *S* versus *(k1, k2)* under the *TAUDT* Scheme.

Lastly Figure 4-9 shows the effect of the two thresholds *(k1, k2)* on the communication overhead per operation (*S*) in *TAUDT*. As *k1* increases, *S* decreases in the key server because aggregating join and leave events reduces the batch rekeying overhead. Similar to what we have observed in Figures 4-7 and 4-8, since there is a small number of untrusted leave requests (*c*), *S* is insensitive to increasing *k2*. Figures 4-9 allows us to find the optimal *(k1, k2)* when given constraints in terms of *D* and P_v. For example, when $D = 5\ s$ and $P_v = 5\%$, the optimal setting *(k1, k2)* is (16, 1) corresponding to the optimal batch rekey interval, $T = 8.83\ s$. The translation of the optimal (*k1, k2*) to the optimal *T* is through the use of Equation 4-10 when evaluating the Petri net model for *TAUDT* discussed earlier.

4.3.3 Join and Leave Double Threshold-based (JALDT) Batch Rekeying

JALDT has two thresholds, *k1* and *k2*, with *k1* checking against the number of join requests *(a)* and *k2* checking against the number of leave requests *(b+c)*, respectively.

Figure 4-10 shows the effect of changing *k1* and *k2* on P_v in *JALDT*. We see that as either *k1* or *k2* increases, P_v increases. The reason is that a higher threshold in either *k1* or *k2* brings more states until rekeying is performed, thus contributing to more states in which the secrecy requirement is violated. Different from the previous results for *TAUDT* (Figures 4-7, 4-8, and 4-9), we observe more distinctions between curves as *k2* increases because *k2* in the *JALDT* protocol is used as a threshold to check against trusted and untrusted leave requests *(b+c)*, not just the number of untrusted requests *(c)* as in *TAUDT*.

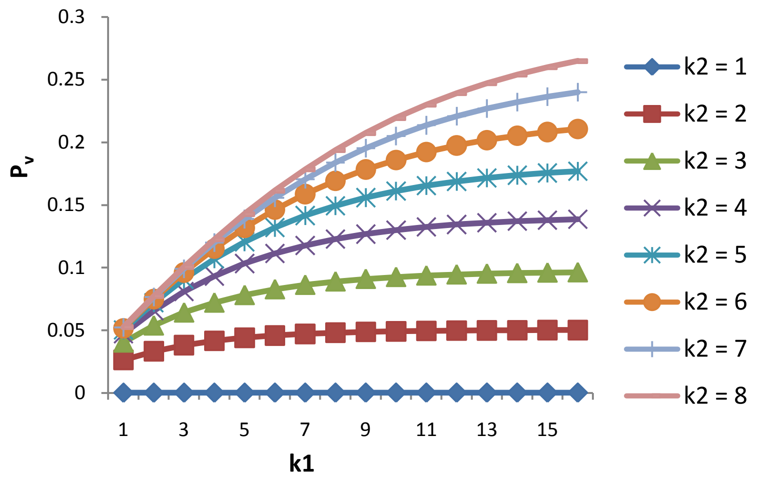


Figure 4- 10: P_v versus *(k1, k2)* under the *JALDT* Scheme.

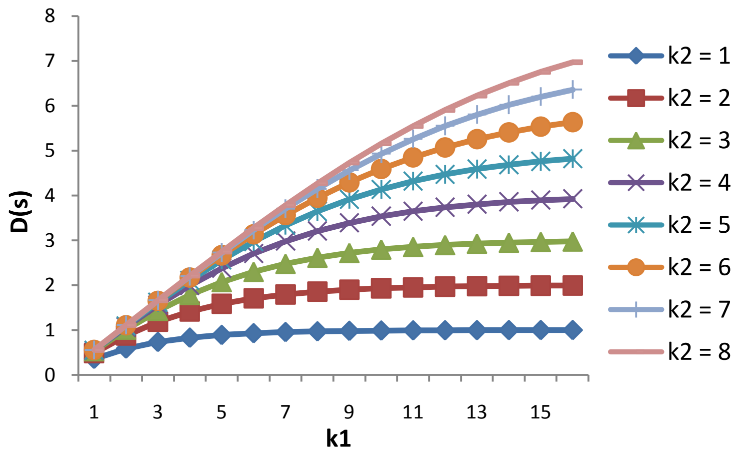


Figure 4- 11: *D* versus *(k1, k2)* under the *JALDT* Scheme.

Figure 4-11 shows the effect of increasing *k1* and *k2* on *D* in *JALDT*. We see that as *k1* and *k2* increase, *D* also increases. Again, using higher thresholds introduces more requests accumulated in the key server until a rekeying operation is performed, resulting in the delay being increased. Also, because there are more leave requests governed by *k2* in the key server, *D* increases as *k2* increases.

Figure 4-12 shows the change of average communication overhead per join/leave operation (*S*) over increasing *k1* and *k2*. Again as *k1* and *k2* increase, *S* decreases because aggregating more join and leave events for a batch rekeying operation will amortize the cost per operation.

From Figures 4-10, 4-11 and 4-12, we can easily determine the optimal (*k1*, *k2*) that would minimize *S* while satisfying *D* and P_v. For example, when D = 5 *s* and P_v = 5%, the optimal setting (*k1*, *k2*) found is (13, 2) corresponding to the optimal batch interval, T = 3.96 *s*. The translation of the optimal (*k1*, *k2*) to the optimal *T* is through the use of Equation 4-10 while evaluating the SPN model for *JALDT*.

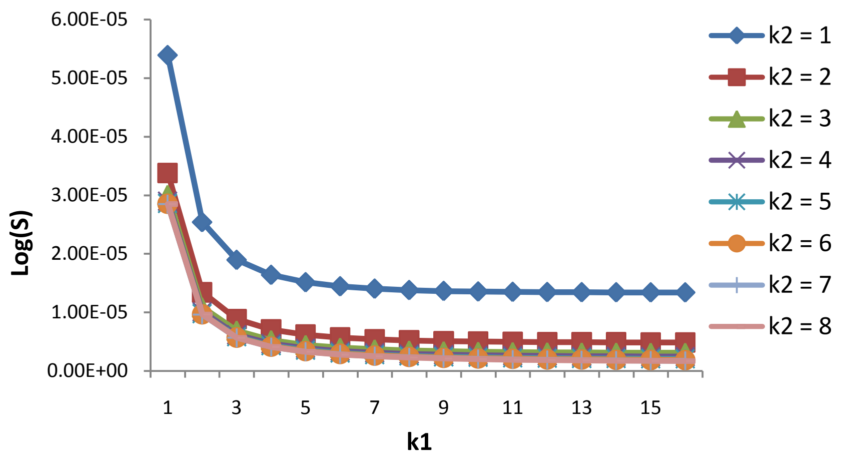


Figure 4- 12: *S* versus (*k1*, *k2*) under the *JALDT* Scheme.

4.3.4 Comparing ULT, TAUDT and JALDT Batch Rekeying Schemes

In this section, we compare the minimum *S* and optimal *T* obtainable while satisfying imposed constraints on *D* and P_v by *ULT*, *TAUDT*, and *JALDT*. We test the sensitivity of our results by varying P_t and λ: μ under identical conditions, and identify the best protocol that minimizes *S* among all while satisfying *D* and P_v.

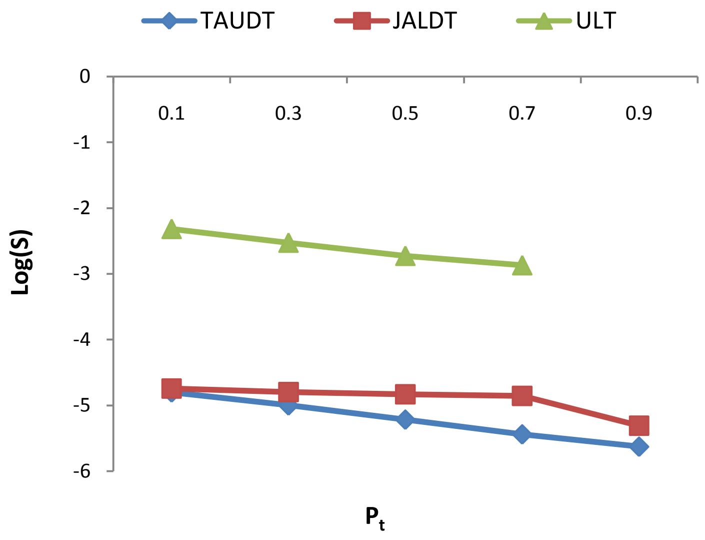


Figure 4- 13: Comparing *ULT*, *TAUDT*, and *JALDT*: *Log (S)* versus P_t.

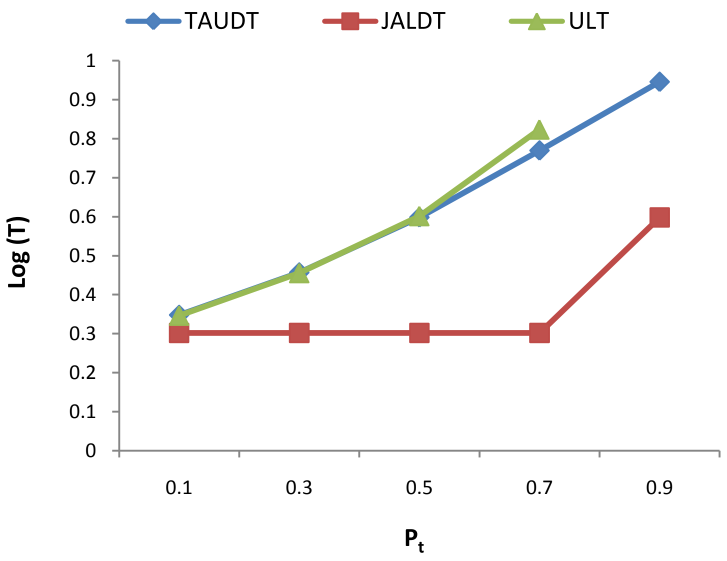


Figure 4- 14: Comparing *ULT*, *TAUDT*, and *JALDT*: *Log (T)* versus P_t.

Figure 4-13 shows the “minimum” *S* (i.e., the optimal *S*) obtained by *ULT*, *TAUDT*, and *JALDT* as a function of P_t. The ratio of $\lambda: \mu$ is set to 1: 0.5 to isolate out its effect. Each data point given is the optimal *S* found satisfying the constraints that *D* = 5 *s* and P_v = 5%. We see that for each curve, as the probability of trustworthiness (P_t) increases, *S* decreases. The reason is that a higher P_t implies less untrusted requests and consequently a lower secrecy violation probability P_v. As a result, a higher P_t reduces the average communication cost per operation. Note that there is no data point at P_t = 0.9 under *ULT* because too much delay has occurred (higher than 5 *s* of *D*) caused by a low arrival rate of $\mu\ (1- P_t)$ for untrusted leave requests in this case.

Among the three threshold-based periodic batch rekeying protocols, *TAUDT* performs the best in terms of the minimum communication cost per join/leave operation (*S*), while satisfying the constraints on *D* and P_v. On the other hand, *ULT* shows the highest minimum *S*. The reason is that *ULT* tends to sharply increase P_v as *k3* increases because *ULT* uses only a single threshold, *k3,* to bound the number of untrusted leave operations. To satisfy the stringent constraint in P_v, *ULT* must select a small *k3* value corresponding to a small optimal *T* and a large minimum *S*. Similarly, since *JALDT* generates a higher P_v than *TAUDT* under identical conditions, *JALDT* has a higher minimum *S* than *TAUDT*. The reason that *JALDT* generates a higher P_v and consequently a higher *S* than *TAUDT* is that *JALDT* has more states having violated forward secrecy because *k2* in *JALDT* checks against *(b+c)* while *k2* in *TAUDT* only checks against *c*.

Figure 4-14 shows the optimal T corresponding to the minimum S found in Figure 4-13. Note that the optimal T at P_v = 0.9 under *ULT* is not available for the same reason that D obtained in this case does not satisfy the constraint.

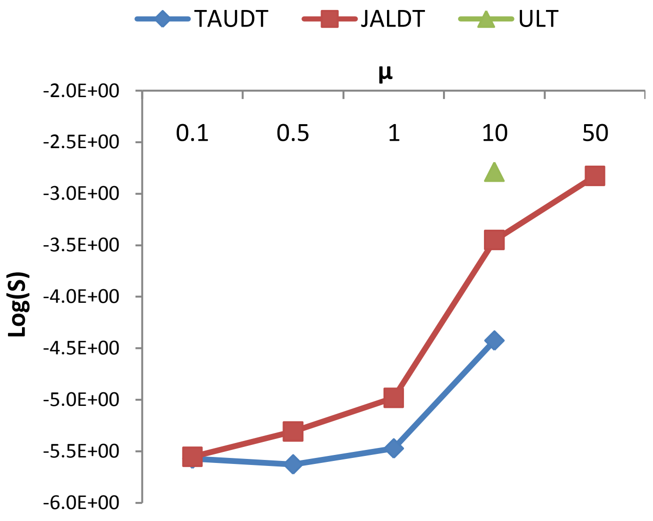


Figure 4- 15: Comparing *ULT*, *TAUDT*, and *JALDT*: *Log (S)* versus *λ: μ*.

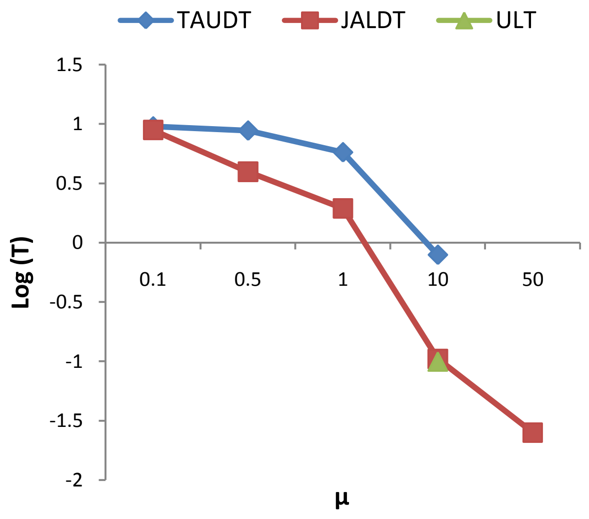


Figure 4- 16: Comparing *ULT*, *TAUDT*, and *JALDT*: *Log (T)* versus *λ: μ*.

Next we compare *ULT*, *TAUDT*, and *JALDT* as a function of the ratio of λ: μ to test the result sensitivity. Figure 4-15 shows the minimum S obtainable when the arrival rate of leave requests (μ) varies (with the arrival rate fixed at 1) to reflect varying ratios of λ: μ. Here we note that the minimum S values at μ = 0.1, 0.5, and 1 in *ULT* are not available because D obtained exceeds the constraint of 5 *s*. The minimum S at μ = 50 under *ULT* could not be found because P_v obtained is too high due to a very high arrival rate of leave requests. Similarly, the minimum S value at μ =

50 under *TAUDT* could not be found because P_v obtained is too high. Figure 4-15 shows that as μ increases, the minimum *S* also increases. The reason is that a higher μ introduces more untrusted leave requests, resulting in a higher P_v, and thus a higher minimum *S*. As discussed earlier, since *TAUDT* is able to generate a lower P_v than *JALDT* under identical conditions, *TAUDT* is able to generate a lower minimum *S*. This is confirmed once again from Figure 4-15, which shows that over a wide range of λ*:* μ ratios, *TAUDT* is the most efficient protocol with the lowest minimum *S* among all.

Finally, we compare all three threshold-based protocols in terms of the optimal *T* corresponding to the minimum *S* found in Figure 4-15. In Figure 4-16, as μ increases, the optimal *T* decreases. Comparing Figure 4-16 with Figure 4-15, whenever there is a minimum *S*, correspondingly there is an optimal *T* generated. We observe that Figure 4-16 correlates well with Figure 4-15 in terms of the trend shown. Specifically, *TAUDT* has the highest optimal *T*, as shown in Figure 4-16, as it has the lowest minimum *S*, as shown in Figure 4-15. Further, *JALDT* shows the second highest optimal *T*, followed by *ULT* which is the last although there is only one value at μ = 10 under *ULT*. Based on Figure 4-16, we conclude that *TAUDT* has the longest optimal *T* compared with the two other threshold-based protocols, by reducing the batch rekeying overhead more efficiently.

4.4 Summary

In this chapter, we investigated a class of QoS-aware threshold-based periodic batch rekeying protocols with the objective of reducing the communication overhead per join/leave operation (*S*) while satisfying delay (*D*) and secrecy (P_v) requirements for wireless GCSs where a centralized key server exists. We have developed performance models to analyze these batch rekeying protocols, and compare their performance characteristics. We observed that an optimal batch rekey interval (*T*) exists under each of these protocols. Further, by varying the probability of trustworthiness among receiving requests (P_t) and the ratio of the arrival rate of join requests to the arrival rate of leave requests (λ*:* μ) over a wide range, we concluded that among the threshold-based periodic batch rekeying protocols proposed (*ULT*, *TAUDT*, and *JALDT*), *TAUDT* is able to produce the minimum *S* and the maximum *T*, which makes it the most efficient protocol among all three. As P_t increases, we observed a decreasing minimum *S* and an increasing *T*. As μ increases, we observed an increasing minimum *S*, and a decreasing optimal *T*.

These results can be used by system designers to determine the optimal *T* value that would minimize *S* under *TAUDT*.

These results reported here are based on a wireless network environment with infrastructure support in which a centralized key server exists for key generation and management. The design and optimization principles of threshold-based periodic batch rekeying can be applied to wireless networks without infrastructure support, i.e., MANETs, as well. We have tested GDH (explained in Chapter 3) as the rekeying protocols and observed the same general trends reported here, with the only difference being the extra delay introduced due to the higher computational complexity of GDH. Later in Chapter 6 when we integrate QoS-aware IDS protocols with threshold-based periodic batch rekeying to deal with both insider and outsider attacks, we will use GDH as the rekeying protocol as we apply threshold-based periodic batch rekeying design optimization principles to MANET environments.

Chapter 5 QOS-AWARE INTRUSION DETECTION FOR MOBILE GROUP COMMUNICATION SYSTEMS

In this chapter, we design and analyze a class of QoS-aware distributed intrusion detection protocols to deal with *insider attacks* in MANET environments. The novelty lies in the analysis of optimal settings in the system design that allows the system to dynamically determine the optimal intrusion detection interval to maximize *MTTSF* of the system while still satisfying performance requirements. The content of this chapter is largely based on our work published in [25-27].

Section 5.1 describes the background of intrusion detection in MANETs and introduces design concepts of our proposed QoS-aware intrusion detection protocols. Section 5.2 discusses the security/performance metrics used, the mathematical model developed for performance evaluation, how the main design parameters are parameterized, and how performance measurements are calculated. Section 5.3 presents numerical results obtained from our analytical model. In particular, it analyzes the effect of main design parameters, compares the proposed intrusion detection protocol against existing intrusion detection protocols, and discusses how our proposed protocols may be applied in practice. Section 5.4 performs a simulation study to validate analytical results obtained. Finally Section 5.5 summarizes this chapter.

5. 1 Distributed Intrusion Detection Protocols

To deal with insider attacks by compromised nodes within the system, we develop distributed intrusion detection protocols employed in GCSs for detecting and evicting compromised nodes. A compromised node in a group can compromise the security of the whole system when useful information has been leaked out to the compromised node. Compromised nodes can also collude to affect faulty decision making. Thus, to tolerate intrusion and maintain the integrity of the system, it is essential to detect and evict compromised nodes. This dissertation research aims to design and analyze QoS-aware distributed intrusion detection protocols and their effect on security and performance characteristics of the resulting GCS. While intrusion detection systems (IDS) for wired networks have been extensively studied, there has been little work on IDS for wireless mobile environments, particularly for MANETs. To evaluate both security and performance characteristics of IDS in GCSs, the approach used in this dissertation research has its root on model-based quantitative analysis [71]. In the literature, not much work has been done in extending model-based quantitative analysis to security analysis. Early work in security emphasizes the prevention of attacks in systems. We believe that the definitions and designs of security properties should reflect specific network and workload environments and take both security and performance requirements into consideration for the optimization of the system.

5.1.1 Background

A mobile wireless network has high security vulnerability because of open medium, dynamic changing network topology, decentralized decision-making and cooperation, lack of centralized authority, lack of resources in mobile devices, and no clear line of defense [12, 67, 112, 113]. Three types of actions against attacks can be taken, namely, prevention, detection, and recovery. Intrusion prevention techniques (e.g., encryption or authentication) can be employed in wireless mobile networks to reduce intrusion. However, security holes cannot be perfectly eliminated [112]. Thus, intrusion detection protocols are introduced as a second line of defense and have become essential for systems with the goal of high-survivability and availability [112].

A compromised node in a group can compromise the security of the whole system when useful information has been leaked out to the compromised node. Compromised nodes can also collude to affect faulty decision making. Thus, to tolerate intrusion and to maintain the integrity of the system, it is essential to detect and evict compromised nodes. In this chapter, we develop a class of QoS-aware intrusion detection protocols employed in GCSs in MANETs for detecting and evicting compromised nodes. Our protocols are QoS-aware in the design in that they allow the system to trade security off for performance, or vice versa, at runtime dynamically to satisfy both the security and performance requirements of the system.

5.1.2 Protocol Description

We consider two types of intrusion detection protocols applicable to GCSs in MANETs, i.e., *host-based IDS* versus *voting-based IDS*. The first type is ***host-based IDS*** [112, 113] for local detection in which each node performs local detection to determine if a neighboring node has been compromised. Each node may implement its host-based IDS preinstalled with standard existing IDS techniques such as misuse detection (also called signature-based detection) and anomaly detection [18, 52, 53, 112, 113] so that our proposed voting-based IDS can be independent of the host-based IDS used as a general framework. Each node may evaluate its neighbors based on information collected, mostly route-related and traffic-related information [39]. We measure the effectiveness of IDS techniques applied (e.g., misuse detection or anomaly detection) for host-based IDS by two parameters, namely, the false negative probability (*p1*) and false positive probability (*p2*). In general, when the system uses misuse detection for IDS, it tends to have more false negatives and less false positives (e.g., higher *p1* and lower *p2*). On the other hand, when the system employs anomaly detection for IDS, it is likely to have fewer false negatives and more false positives (e.g., lower *p1* and higher *p2*).

The second type is ***voting-based IDS*** for cooperative detection based on majority voting. Voting-based IDS derives from the fault tolerance concept based on majority voting for evicting a target node in the context of sensor networks [17]. For voting-based IDS to be performed, each node is preinstalled with host-based IDS to collect information to detect the status of neighboring nodes. Periodically a target node would be evaluated by *m* vote-participants dynamically selected where *m* is a design parameter. If the majority of *m* nodes decided to vote against the target node, then the target node would be evicted from the system by means of rekeying. This adds intrusion tolerance to tolerate collusion of compromised nodes in MANETs as it takes the majority of "bad" nodes among *m* nodes to work against the system. We characterize voting-based IDS by two parameters, namely, false negative probability (P_{fn}) and false positive probability (P_{fp}). These two parameters can be calculated based on (a) the *per-node* false negative and positive probabilities (*p1* and *p2*) of host-based IDS in each node; (b) the number of vote-participants, *m*, selected to vote for or against a target node; and (c) an estimate of the current number of compromised nodes which may collude with the objective to disrupt the service of the system. Since *m* nodes are selected to vote, if the majority of *m* voting-participants (i.e., $\geq \lceil m/2 \rceil$) cast negative votes against a target node, the target node is regarded as compromised and will be evicted from the system.

5.2 Performance Analysis

5.2.1 Metrics

We use *MTTSF* to measure the security property and $\hat{C}_{total}$ to measure the performance property of secure GCSs in MANETs as follows:

• ***MTTSF*** **(Mean Time to Security Failure)**: This metric indicates the average time elapsed for the GCS to reach a security failure state. *MTTSF* can also be viewed as the system lifetime metric. The GCS fails when any mobile group contained within reaches a security failure state when (1) data have been leaked out to a compromised but undetected member node (i.e., C1), or (2) more than *1/3* of the member nodes have been compromised (i.e., C2). Note that illegal data leak-out only occurs when a compromised but undetected member requests data and subsequently obtains data using the group key. As a security metric, lower *MTTSF* means faster *loss of system integrity* or *loss of availability*. A design goal is to identify the optimal intrusion detection interval to maximize *MTTSF*.

• $\hat{C}_{total}$ **(Communication Traffic Cost)*:*** This metric indicates total traffics incurred per time unit (*s*) including group communication, status exchange, rekeying, intrusion detection, beacon, group partition/merge and mobility-induced activities. Since all nodes share a wireless bandwidth *BW*, a high $\hat{C}_{total}$ will be translated into a high level of contention and consequently a high query response time for group communication. A design goal is to minimize $\hat{C}_{total}$.

5.2.2 Performance Model

We develop a Stochastic Petri net (SPN) model as shown in Figure 5-1 to describe the behavior of a mobile group in the presence of insider attacks and intrusion detection activities, with the goal of identifying optimal intrusion detection intervals to maximize *MTTSF* while satisfying imposed performance requirements. Table 5-1 summarizes places and transitions with their physical meanings, along with transition rates, arcs, and arc multiplicities in the SPN model.

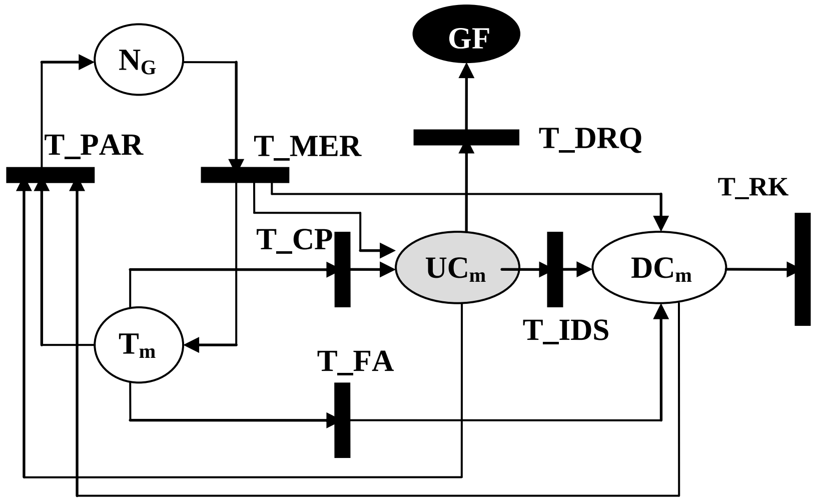


Figure 5- 1: SPN Model of a Mobile Group in the GCS.

Table 5- 1: Places, Transitions, Transition Rates, Arcs and Arc Multiplicities for SPN Model in Figure 5-1.

Place	Meaning		
T_m	*mark(T_m)* means the number of trusted member nodes		
UC_m	*mark(UC_m)* means the number of compromised but undetected member nodes		
DC_m	*mark(DC_m)* means the number of compromised and detected member nodes		
GF	*mark(GF)* == 1 means that group security failure has occurred due to illegal data leak-out		
Transition	**Rate**	**Physical Meaning**	
T_CP	$A\ (m_c)$	A node has been compromised	
T_IDS	$mark(UC_m)^*\ D\ (m_d)^*(1-P_{fn})$	A compromised node has been detected	
T_FA	$mark(T_m)^*\ D\ (m_d)^*P_{fp}$	A node has been falsely diagnosed as compromised	
T_RK	$1/T_{cm}$	A rekeying operation has been performed	
T_DRQ	$mark(UC_m)^*p1^*\lambda_q$	A group communication operation has been performed	
Input arc	**Multiplicity**	**Output arc**	**Multiplicity**
T_m –*T_CP*	1	*T_CP* – UC_m	1
T_m –*T_FA*	1	*T_FA* – DC_m	1
UC_m –*T_IDS*	1	*T_IDS* – DC_m	1
DC_m –*T_RK*	1	*T_DRQ* – UC_m	1
UC_m –*T_DRQ*	1	*T_DRQ* – *GF*	1

Here we describe how the SPN model is constructed as follows:

- The SPN model describes the behavior of a *single* mobile group as it evolves. This mobile group may partition into two and may merge with another group during its lifetime. We track trusted members, compromised members undetected, and compromised members detected during the group's lifetime to understand its security and performance characteristics. Certainly the system knows the number of compromised nodes detected by IDS at all times. However, the system does not know the number of compromised nodes not yet detected. It only knows the total number of member nodes. The SPN model predicts the number of compromised but yet detected nodes through knowledge of the node compromising rate or the attacker function explained below.
- We use places to classify nodes except for place N_G which holds the current number of groups in the system. Specifically, place T_m holds trusted members, UC_m holds compromised nodes not yet detected by IDS, and DC_m holds compromised nodes that have been detected by IDS. Note that T_m, UC_m, and DC_m represent nodes in one group, not in the system. To be more specific, the numbers of nodes in places T_m, UC_m, and DC_m, obtained by $mark(T_m)$, $mark(UC_m)$, and $mark(DC_m)$, respectively, would be adjusted based on the number of groups existing in the system (obtained by $mark(N_G)$), which changes upon group merge/partition events.
- We use transitions to model events. Specifically, *T_MER* and *T_PAR* model the group merge or partition events, respectively; *T_CP* models a node being compromised. *T_FA* models a node being falsely identified as compromised. *T_IDS* models a compromised node being detected. *T_RK* models rekeying. *T_DRQ* models a data leak security failure (i.e., C1). A firing of a transition will change the state of the system, which is represented by the distribution of tokens in the SPN. For example, $mark(N_G)$ changes upon firing *T_MER* or *T_PAR* since the number of groups changes upon a group merge or partition event; the number of compromised nodes undetected increments by 1 and, place UC_m will hold one more token when *T_CP* fires. A transition is eligible to fire when the firing conditions associated with the event are met. The firing conditions are (1) its input place must contain at least one token and (2) the associated enabling guard function, if exists, must return *true*. For example, *T_CP* is enabled to fire when there exists "good" nodes in the group, that is, place T_m holds at least one token, and the enabling function associated with *T_CP* returns *true*.

- Except for tokens contained in place N_G, we use a "token" in the SPN model to represent a node in the group. The population of each type of nodes is equal to the number of tokens in the corresponding place. Initially, all N members are trusted in one group and put in place T_m as tokens.
- Trusted members may become compromised because of insider attacks with a node-compromising rate $A(m_c)$. This is modeled by firing transition *T_CP* and moving tokens one at a time (if it exists) from place T_m to place UC_m. See Equation 5-4 for the parameterization of $A(m_c)$.
- Tokens in place UC_m represent compromised but undetected member nodes. We consider the system as having experienced a security failure when data are leaked out to compromised but undetected members, i.e., C1. A compromised and undetected member will attempt to compromise data from other members in the group. Because of the use of host-based IDS, a node will reply to such a request only if it could not identify the requesting node as compromised with the false negative probability *p1*. This is modeled by associating transition *T_DRQ* with rate $p1*\lambda_q * mark\ (UC_m)$. The firing of transition *T_DRQ* will move a token into place *GF*, at which point we regard the system as experiencing a security failure due to C1.
- A compromised node in place UC_m may be detected by IDS before it compromises data in the GCS. The intrusion detection activity of the mobile group is modeled by the IDS detection rate $D(m_d)$. See Equation 5-5 for the parameterization of $D(m_d)$. Whether the damage has been done by a compromised node before the compromised node is detected depends on the relative magnitude of the node-compromising rate ($A(m_c)$) versus the IDS detection rate ($D(m_d)$). When transition *T_IDS* fires, a token in place UC_m will be moved to place DC_m, meaning that a compromised but undetected node now becomes detected by IDS. For voting-based IDS, the transition rate of *T_IDS* is $mark(UC_m)*D(m_d)*\ (1-P_{fn})$, taking into consideration of the number of compromised but yet detected nodes and the false negative probability of voting-based IDS. Voting-based IDS can also false-positively identify a trusted member node as compromised. This is modeled by moving a trusted member in place T_m to place DC_m after transition *T_FA* fires with rate $mark(T_m)*D(m_d)*\ P_{fp}$. Note that voting-based IDS parameters, P_{fn} and P_{fp}, can be derived based on *p1* and *p2*, the number of vote-participants (*m*), and the current number of compromised nodes which may collude to disrupt the service of the system. Later we will show how we may parameterize P_{fn} and P_{fp}.

- Finally, the mobile group experiences a security failure if either security failure condition, C1 or C2, is met. We model this by making the group enter an absorbing state when either C1 or C2 is *true*. To achieve this, we associate every transition in the SPN model with an enabling function that returns *false* (disabling the transition from firing) when either C1 or C2 is met, and returns *true* otherwise. For the SPN model, C1 is *true* when *mark(GF)* > 0 representing that data have been leaked out to compromised, undetected members; C2 is *true* when more than 1/3 of member nodes are compromised but undetected as indicated by:

$$\frac{mark(UC_m)}{mark(T_m)+mark(UC_m)} > \frac{1}{3} \qquad (5\text{-}1)$$

5.2.3 Parameterization

Here we describe the parameterization process, i.e., how to give model parameters proper values reflecting the operational and environment conditions of the system.

- ***Λ_{J+L}*:** This indicates the join and leave rates of all current nodes in equilibrium. For efficiency and simplicity, we implicitly model the join and leave events in our SPN model. Assume that a node may leave the group voluntarily with rate μ and may rejoin the group with rate λ due to tactical reasons. Then, the probability that a node is in the group is $\lambda/(\lambda+\mu)$ and the probability that it is not is $\mu/(\lambda+\mu)$. It follows that the average number of nodes that are active members is given by $N*\lambda/(\lambda+\mu)$, where N is the number of current members. Furthermore, let Λ_{J+L} be the join and leave rate of all current nodes in equilibrium and can be calculated as:

$$\Lambda_{J+L} = \left[\lambda \times N \times \frac{\mu}{(\lambda+\mu)}\right] + \left[\mu \times N \times \frac{\lambda}{(\lambda+\mu)}\right] \qquad (5\text{-}2)$$

Since N represents the number of current member nodes in a group, $N = mark\,(T_m) + mark(UC_m)$ where $mark\,(T_m)$ returns the number of trusted nodes and $mark\,(UC_m)$ returns the number of undetected compromised nodes in a group. Note that N includes $mark\,(UC_m)$ since the system regards nodes compromised but undetected by IDS as trusted. The number of tokens initially placed in T_m is $N = N_{init} * \lambda/(\lambda+\mu)$.

- ***T_{cm}*:** Recall that T_{cm} is the communication time required for broadcasting a rekey message for a join or leave event. The reciprocal of T_{cm} is the rate of transition *T_RK*. Based on GDH, the following formula calculates T_{cm}:

$$\begin{aligned} & if(N > 1) \qquad\qquad \text{(5-3)} \\ & T_{cm} = \frac{3b_{GDH}(N-1)}{BW} \\ & else \\ & T_{cm} = \frac{b_{GDH}}{BW} \end{aligned}$$

where N is the number of current member nodes, b_{GDH} is the length of an intermediate value, and BW is the wireless network bandwidth (*Mbps*). We assume that the size of the rekey message is at least B_{GDH} when the current number of members is zero or one.

• ***A* (*m*$_c$):** This is an attacker function that returns the rate at which nodes are compromised in the mobile group. It will apply to transition *T_CP* in the SPN model. Three different attacker functions are considered based on the time taken to compromise a node, namely, *logarithmic time attacker, linear time attacker*, and *polynomial time attacker*, as follows:

$$A_{log}(m_c) = \lambda_c \times log_p(m_c) \qquad \text{(5-4)}$$

$$A_{linear}(m_c) = \lambda_c \times m_c$$

$$A_{poly}(m_c) = \lambda_c \times (m_c)^p$$

$$where\ m_c = \frac{mark(T_m) + mark(UC_m)}{mark(T_m)}$$

These three functions differ by the way the node compromising rate increases as more nodes become compromised. For the linear attacker function, the node compromised rate increases linearly with the number of compromised nodes. Hence, $A_{linear}(m_c) = \lambda_c m_c$ where m_c reflects the degree of compromised nodes currently in the group and λ_c is the base node compromising rate initially given that there is no compromised node in the group. For $A_{log}(m_c)$, the compromising rate increases in logarithm form with the number of compromised nodes. For $A_{poly}(m_c)$ the compromising rate increases in exponential form with the number of compromised nodes. Note that these three forms are prediction functions for the node compromising rate. In practice, one would observe the number of compromised nodes over a time period and time points at which these compromised nodes are detected and apply curve-fit techniques to know which function reflects the attacker behavior. We also note that p is a base index parameter selected to reflect the degree of changes of the logarithmic and polynomial attacker functions with respect to the number of compromised nodes. It requires a fine tune after sufficient data are collected. We choose *p*=3.

• ***D* (*m*$_d$)**: This is a detection function that returns the rate at which IDS is invoked. Three different detection functions, namely, *logarithmic periodic detection*, *linear periodic detection*, and *polynomial periodic detection*, are parameterized as follows:

$$D_{log}(m_d) = \frac{1}{T_{IDS}} \times log_p(m_d) \tag{5-5}$$

$$D_{linear}(m_d) = \frac{1}{T_{IDS}} \times m_d$$

$$D_{poly}(m_d) = \frac{1}{T_{IDS}} \times (m_d)^p$$

$$where\ m_d = \frac{N_{init}}{mark(T_m) + mark(UC_m)}$$

These three functions differ by the way the detection rate changes with the number of compromised nodes that have been detected by IDS. For the linear detection function, the IDS detection rate increases linearly with the number of compromised nodes detected. D_{linear} *(m_c)* is the linear periodic detection function where m_c indicates the degree of compromised nodes that have been detected by IDS, and T_{IDS} is the base detection time interval which we aim to determine for maximizing *MTTSF* when applying voting-based IDS. The log detection function, D_{log} *(m_d)*, and exponential detection function, D_{poly} *(m_d)*, have the same form as their counterparts in the attacker function. We note again that *p* is a base index parameter selected to reflect the degree of changes of the logarithmic and polynomial detection functions with respect to the number of compromised nodes detected. We again choose *p*=3.

P_{fp} *or* P_{fn} (5-6)

$$= \sum_{i=0}^{m-N_{majority}} \left[\frac{C\binom{N_{bad}}{N_{majority}+i} \times C\binom{N_{good}}{m-(N_{majority}+i)}}{C\binom{N_{good}+N_{bad}}{m}} \right]$$

$$+ \sum_{i=0}^{m-N_{majority}} \left[\frac{C\binom{N_{bad}}{i} \times \sum_{j=N_{majority}-i}^{m-i} \left[C\binom{N_{good}}{j} \times p^{j} \times C\binom{N_{good}-j}{m-i-j} \times (1-p)^{(m-i-j)} \right]}{C\binom{N_{good}+N_{bad}}{m}} \right]$$

$$where\ N_{majority} = \left\lceil \frac{m}{2} \right\rceil, N = N_{good} + N_{bad} = mark(T_m) + mark(UC_m)$$

p=per-node false negative prob.(p1) for P_{fn} *or*

per-node false positive prob.(p2) for P_{fp}

m = number of vote-participants

Note that $C\binom{n}{k} = 0\ if\ n < k$

• ***P****fn* **&** ***P****fp***:** P_{fn} is the probability of false negatives defined as the number of compromised nodes diagnosed by voting-based IDS as trusted healthy nodes (i.e., detecting a bad node as a good node) over the number of detected nodes. On the other hand, P_{fp} is the probability of false positives defined as the number of normal nodes flagged as anomaly over the number of trusted normal nodes. We consider the intrinsic defect of host-based IDS in each node as well as the possible collusion of compromised nodes during the voting process. For example, if a vote-participant is compromised, it can cast a negative vote to evict a healthy target node in the group or it can cast a positive vote for a malicious node to keep more compromised nodes in the group. Equation 5-6 reflects these two cases of false positives or false negatives introduced into the group respectively. In Equation 5-6, *m* is the number of vote-participants to cast a vote against a target node, N_{bad} is the number of currently compromised nodes in the group represented as *mark* (UC_m), and N_{good} is the number of currently healthy nodes in the group indicated as *mark* (T_m). Accordingly, P_{fp} is obtained when the majority of voters consists of bad nodes who cast a negative vote against a good node, and good nodes who mistakenly diagnose a good node as a bad node with the probability of *p2* (i.e., *p2* is a per-node false positive probability), resulting in a healthy node being evicted. On the other hand, P_{fn} occurs when the majority of voters is from positive votes by bad nodes (i.e., casting a positive vote against a bad node) or good nodes who mistakenly diagnose a bad node as a good node with the probability of *p1* (i.e., *p1* is a per-node false negative probability), keeping more compromised nodes undetected in the group. Note that as default, even though we use *p1* = *p2* = 1% as fixed probability, which deems acceptance in most existing IDS protocols used in practice, P_{fn} and P_{fp} are constantly being adjusted to properly react to dynamically changing network and operational conditions, such as the degree of compromised nodes, node density, and number of vote-participants (*m*) used over time.

• **Group Merge and Partition:** We model group merge and partition events by a *birth-death process* with arrival rate = $\lambda_{np,i}$ and departure rate = $\mu_{nm,i}$ where state *i* represents that there are *i* mobile groups in the GCS. We obtain *group* merging/partitioning rates as follows. We first observe the number of merge and partition events by simulation for a sufficiently long period of time *T*. We next observe the sojourn time S_i in state *i*, i.e., when *i* groups are present in the system. Let $N_{nm,i}$ and $N_{np,i}$ be the numbers of group merge or partition events observed in state *i*, respectively. Then, the merging/partitioning rates in state *i*, represented by $\mu_{nm,i}$ and $\lambda_{np,i}$, are computed by the first-order approximation as:

$$\mu_{nm,i} = \frac{N_{nm,i}}{S_i} \qquad \lambda_{np,i} = \frac{N_{np,i}}{S_i} \qquad \text{(5-7)}$$

Note that the group merging/partitioning rates parameterized above are a function of the node mobility and density in general. We observe that when node density is high, group merge is more likely to occur than group partition, leading to a smaller number of groups (lower *i*) observed in the system. On the other hand, as the node density is low, the system is more likely to stay at large number of groups (higher *i*) with high probability. In other words, when the node density is low, group partition is more likely to occur than group merge.

5.2.4 Calculations of Metrics

MTTSF can be obtained by calculating the *mean time to absorption* (*MTTA*) of the SPN model through assigning proper rewards to states of the system [84]. We use a different reward assignment to calculate *MTTSF* under SF1 and SF2 system failure definitions. *MTTSF* under SF1 is calculated by assigning a reward of 1 to all states except for absorbing states in which C1 or C2 is met. We do this reward assignment because the system fails when any single group fails. Recall that the SPN model developed is for modeling the lifetime of a single group. On the other hand, *MTTSF* under SF2 is calculated by assigning a reward of:

$$r_i = \left(\frac{1}{n} + \cdots + \frac{1}{1}\right) \qquad \text{(5-8)}$$

to all states except for the absorbing states in which C1 or C2 is met. We do this assignment because the system fails when all groups fail. Thus based on the concept of the *mean time to failure* of a *1-out-of-n* system [84] where *n* is the number of groups in the GCS given by *mark*(N_G), we would accumulate a reward of ($1/n + \ldots + 1$) instead of just 1 toward the system lifetime in those states in which the system is still alive.

After proper rewards to states are assigned as above, the *MTTSF* of the GCS can be calculated by the expected accumulated reward until absorption, $E[Y(\infty)]$, defined as:

$$E[Y(\infty)] = \sum_{i \in S}^{\infty} r_i \int_0^{\infty} P_i(t)dt \qquad (5\text{-}9)$$

where *S* denotes the set of all states except the absorbing states and $P_i(t)$ is the probability of state *i* at time *t*. For all *i* states, $r_i = 1$ under SF1, and r_i is given by Equation 5-8 under SF2.

We calculate $\hat{\boldsymbol{C}}_{\boldsymbol{total}}$ by the probability-weighted average of $\hat{C}_{total,i}$ representing the communication cost incurred per time unit (*s*) in state i. Specifically, $\hat{C}_{total}$ is calculated by accumulating $\hat{C}_{total,i}(t)$ over *MTTSF* divided by *MTTSF*, i.e.,

$$\hat{C}_{total} = \frac{\int_0^{MTTSF} \hat{C}_{total,i}(t)dt}{MTTSF} \qquad (5\text{-}10)$$

$\hat{\boldsymbol{C}}_{\boldsymbol{total,i}}$ is calculated as:

$$\hat{C}_{total,i} = \hat{C}_{GC,i} + \hat{C}_{status,i} + \hat{C}_{rekey,i} + \hat{C}_{IDS,i} + \hat{C}_{beacon,i} + \hat{C}_{mp,i} \qquad (5\text{-}11)$$

where $\hat{C}_{GC,i}$, $\hat{C}_{status,i}$, $\hat{C}_{rekey,i}$, $\hat{C}_{IDS,i}$, $\hat{C}_{beacon,i}$, and $\hat{C}_{mp,i}$, are the cost components for group communication, status exchange, rekeying, intrusion detection, beacon, group partition/merge, and mobility events, respectively, given that the number of groups in the system is *i*. Below we explain how we calculate $\hat{C}_{GC,i}$, $\hat{C}_{status,i}$, $\hat{C}_{rekey,i}$, $\hat{C}_{IDS,i}$, $\hat{C}_{beacon,i}$ and $\hat{C}_{mp,i}$. Note that we have omitted (*t*) in each term of Equation 5-11 for simplicity.

• $\hat{\boldsymbol{C}}_{\boldsymbol{GC,i}}$**:** this cost includes the communication cost incurred by group communication activities. It is calculated by:

$$\hat{C}_{GC,i} = \lambda_q \times N \times b_{GC} \times H \qquad (5\text{-}12)$$

where λ_q is the group communication rate, *N* is the number of active group members in the single group we are observing (i.e., *mark* (UC_m)+ *mark*(T_m)), b_{GC} is the message size (*bits*) of a group communication packet, and H is the number of hops a multicast packet travels from a node to all group members connected by a binary tree structure, given as:

$$H = {}^{r_i}/_{R} \times (N-1) \;\; where \;\; r_i = {}^{r}/_{\sqrt{i}} \qquad \text{(5-13)}$$

Here r_i represents the radius of the operational group area where i indicates the number of groups existing in the system, and R refers to the wireless radio range used.

• $\hat{C}_{status,i}$**:** this cost is for group node *status exchange* for intrusion detection. It is calculated by:

$$\hat{C}_{status\,,i} = \frac{(N \times b_s) \times H}{T_{status}} \qquad \text{(5-14)}$$

where T_{status} is the periodic time interval for disseminating a status exchange message, N is the number of group members, and b_s is the message size (*bits*) of the status exchange information.

• $\hat{C}_{rekey,i}$**:** this cost is for group key rekeying due to join/leave events and forced evictions to evict detected compromised nodes. It is calculated as:

$$\hat{C}_{rekey\,,i} = \hat{C}_{join\,/leave\,,i} + \hat{C}_{eviction\,,i} \qquad \text{(5-15)}$$

where $\hat{C}_{join/leave,\,i}$ is the cost introduced by leave and join operations per time unit and $\hat{C}_{eviction,\,i}$ is the cost introduced by forced evictions per time unit. The term $\hat{C}_{join/leave,i}$ is calculated by:

$$\hat{C}_{join\,/leave\,,i} = \Lambda_J \times C_{join\,,i} + \Lambda_L \times C_{leave\,,i} \qquad \text{(5-16)}$$

where Λ_J and Λ_L are aggregate join and leave rates as given in Equation 5-2, and $C_{join\,,i}$ and $C_{leave\,,i}$ are the rekeying cost per join/leave operation, calculated as:

$$C_{join\,,i} = C_{leave\,,i} = \left(H \times M_{update}^{members}\right) + C_{GDH,i} \qquad \text{(5-17)}$$

$$C_{GDH,i} = \{b_{GDH}(2N-3) \times (N-1)\} + \{b_{GDH} \times N \times H\} \qquad \text{(5-18)}$$

where H is as given in Equation 5-13, $M_{update}^{members}$ is the number of bits to update the group member view, and $C_{GDH,i}$ is the rekeying cost when i groups exist in the system. In Equation 5-18, the first term indicates the unicast communication cost in stages of 1 and 3 of GDH while the second term accounts for the multicast communication cost in stages of 2 and 4 of GDH. The term $\hat{C}_{eviction,i}$ in Equation 5-15 is calculated as:

$$\hat{C}_{eviction,i} = [rate(T\_IDS) + rate(T\_FA)] \times \hat{C}_{leave,i} \qquad (5\text{-}19)$$

where *rate(T_IDS)* is the IDS intrusion detection rate and *rate(T_FA)* gives the IDS false alarm rate by which nodes are identified as compromised nodes. Both rates may be obtained readily from evaluating the SPN performance model.

- $\boldsymbol{\hat{C}_{IDS,i}}$**:** this is the communication cost due to IDS. For voting-based IDS, this cost is computed as:

$$\hat{C}_{IDS,i} = D(m_d) \times \left(1 - P_{fn}\right) \times N \times [b_{m-list} + m \times b_v] \times H \qquad (5\text{-}20)$$

where *D(*m_d*)* is the detection rate, P_{fn} is the probability of false negatives, *N* is the number of current members in a group, *m* is the number of vote-participants against a target node, $b_{m\text{-}list}$ is the message size (*bits*) of the list containing *m* vote participants, and b_v is the message size (*bits*) of a vote.

- $\boldsymbol{\hat{C}_{beacon,i}}$**:** this is the communication cost due to beaconing messages being multicast to group members. It is calculated by:

$$\hat{C}_{beacon,i} = \Lambda_{RB} \times M_{alive} \times H \qquad (5\text{-}21)$$

$$\Lambda_{RB} = N \times \left[\frac{\lambda}{\lambda + \mu}\right] \times \frac{1}{T_{RB}} \qquad (5\text{-}22)$$

- $\boldsymbol{\hat{C}_{mp,i}}$**:** this is the communication cost due to group merging and partitioning events. It is computed by:

$$\hat{C}_{mp,i} = C_{partition,i} + C_{merge,i} \qquad (5\text{-}23)$$

$$C_{partition,i} = \lambda_{np,i} \times C_{np,i} \qquad (5\text{-}24)$$

$$C_{merge,i} = \mu_{nm,i} \times C_{nm,i} \qquad (5\text{-}25)$$

$$C_{np,i} = 2 \times \left(H \times M_{update}^{members} + C_{GDH,i}\right) \qquad (5\text{-}26)$$

$$C_{nm,i} = H \times M_{update}^{members} + C_{GDH,i}$$

where $\lambda_{np,i}$ and $\mu_{nm,i}$ are group partitioning and merging rates where there exist i groups in the system, as given in Equation 5-7, $C_{np,\,i}$ and $C_{nm,\,i}$ are communication costs generated by group partition and merge events when the system is in state i, $M_{update}^{members}$ is the number of bits to update the group member view, and H is the number of hops from a node to other group members.

5.3 Numerical Data and Analysis

5.3.1 Optimal Intrusion Detection Interval T_{IDS}

We present numerical data obtained through the evaluation of the SPN model developed and provide physical interpretations. Our objective is to identify optimal intrusion detection interval (T_{IDS}) that will maximize *MTTSF* while satisfying performance requirements of the system. We also identify the best detection function to use in response to the attacker function (representing the node compromising rate) detected at runtime.

Table 5- 2: Parameters and Their Default Values.

Parameter	Values	Parameter	Values	Parameter	Values
λ	once per 1 hr	***p1=p2***	1%	b_v	8 bytes
μ	once per 4 hrs	***r***	500 m	b_{GC}	100 bytes
T_{IDS}	5-1200 (s)	N_{init}	100	b_{GDH}	8 bytes
T_{status}	2 (s)	***D*** (m_d)	Linear attack	***p***	3
λ_c	once per 12 hrs	***A*** (m_c)	Linear detection	***m***	5
λ_q	once per min	b_s	50 bytes	***BW***	1Mbps

Table 5-2 summarizes the set of parameters and their default values used. In particular, we use *p1* = *p2* = 1% since in general less than 1% of false positive or false negative rate is desirable. We note that *p1* and *p2* are two parameters representing false negative and false positive probabilities for characterizing any host-based IDS. We vary the values of selected parameters including the number of vote-participants in voting-based IDS (*m*), group communication rate (λ_q), and base compromising rate (λ_c) to analyze their effects on the optimal base detection interval for maximizing *MTTSF*.

Below we first analyze the effect of intrusion detection interval (T_{IDS}) on *MTTSF* as a function of the number of vote-participants (m) and demonstrate that there exists an optimal intrusion detection interval (T_{IDS}) for maximizing *MTTSF* or minimizing $\hat{C}_{total}$ with proper physical interpretations given. *MTTSF* here is calculated based on SF1 being the system failure definition. Later we will present data for cases where SF2 is the system failure definition.

Below we first analyze the effect of intrusion detection interval (T_{IDS}) on *MTTSF* as a function of the number of vote-participants (m) and demonstrate that there exists an optimal intrusion detection interval (T_{IDS}) for maximizing *MTTSF* or minimizing $\hat{C}_{total}$ with proper physical interpretations given.

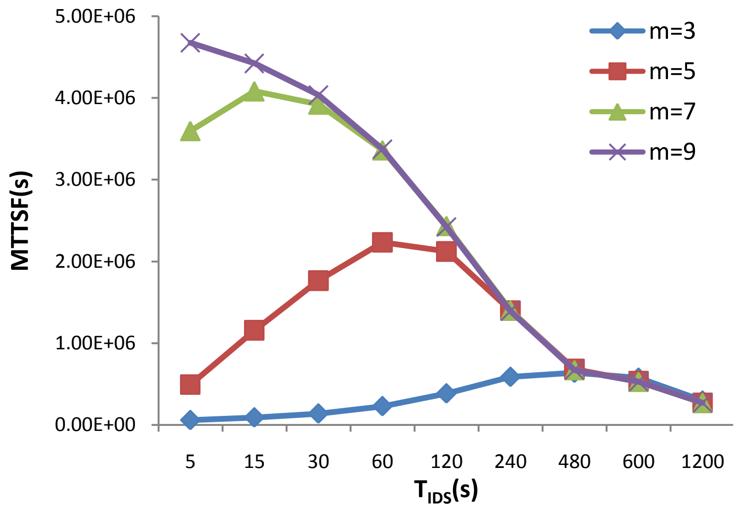


Figure 5- 2: Effect of *m* on *MTTSF* and Optimal T_{IDS}.

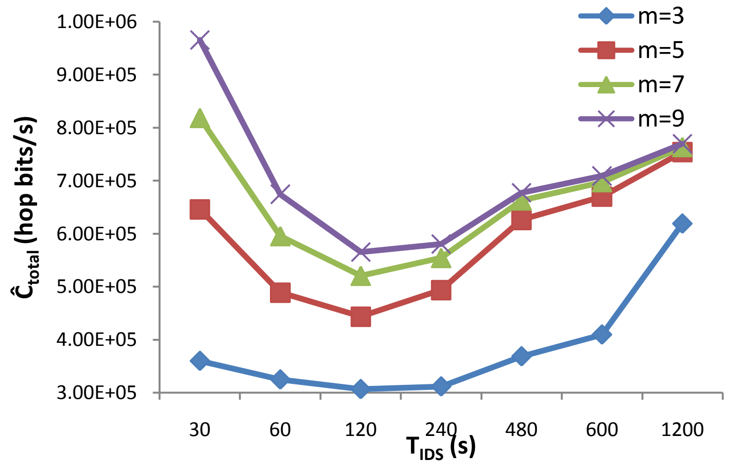


Figure 5- 3: Effect of *m* on $\hat{C}_{total}$ and Optimal T_{IDS}.

Figure 5-2 shows the effect of intrusion detection interval (T_{IDS}) on *MTTSF* as the number of vote-participants (m) changes for the case in which the attacker function and the detection function are both linear. We observe that there exists an optimal T_{IDS} that maximizes *MTTSF* for each given m value. In general, as T_{IDS} becomes larger, *MTTSF* increases until its optimal point reaches, and then *MTTSF* decreases after the optimal point. The reason of increasing *MTTSF* as T_{IDS} increases initially is that as T_{IDS} increases there are fewer nodes being falsely identified by IDS since IDS is triggered less often, thus reducing the system failure probability due to C2. After the optimal T_{IDS} is reached, *MTTSF* decreases again because IDS is not triggered often enough to detect compromised nodes which may perform attacks to cause system failures due to C1. Note that P_{fp} is one aspect of false alarms generated by IDS, and therefore more nodes will be falsely identified as compromised nodes if IDS is more frequently triggered.

Further, we also observe the sensitivity of optimal T_{IDS} identified on *MTTSF* as m varies. When m is large, the false alarm probability ($P_{fa} = P_{fp} + P_{fn}$) is small because more nodes are participating in the voting process, reducing the possibility of collusion by compromised nodes. Consequently, when m is large, we observe a high *MTTSF* due to the small false alarm probability. Conversely, when m is small, *MTTSF* is small due to a larger false alarm probability. A smaller m also results in a longer optimal T_{IDS} being used to maximize *MTTSF* to offset the adverse effect of IDS with large false positives, e.g., optimal T_{IDS} = 480, 60, 15, and 5 s for m = 3, 5, 7, and 9 respectively.

Figure 5-3 shows the overall communication cost ($\hat{C}_{total}$) versus intrusion detection interval (T_{IDS}) as the number of vote-participants (m) varies. An optimal T_{IDS} exists in each curve (minimum $\hat{C}_{total}$) because of the tradeoff between decreasing normal group communication costs ($\hat{C}_{GC,i}$) and increasing IDS related communication costs ($\hat{C}_{eviction,i} + \hat{C}_{IDS,i}$) as T_{IDS} becomes shorter. Also we observe that when m is large, $\hat{C}_{total}$ is high. This is because a larger m induces a lower P_{fp} under which more nodes will be able to perform normal group activities. Furthermore, when there are more vote participants, there is a higher cost associated with dynamic majority voting. Contrary to *MTTSF* versus T_{IDS}, we do not observe the sensitivity of an optimal T_{IDS} identified, but there is a relatively higher communication cost saved when the optimal T_{IDS} identified is employed as m increases.

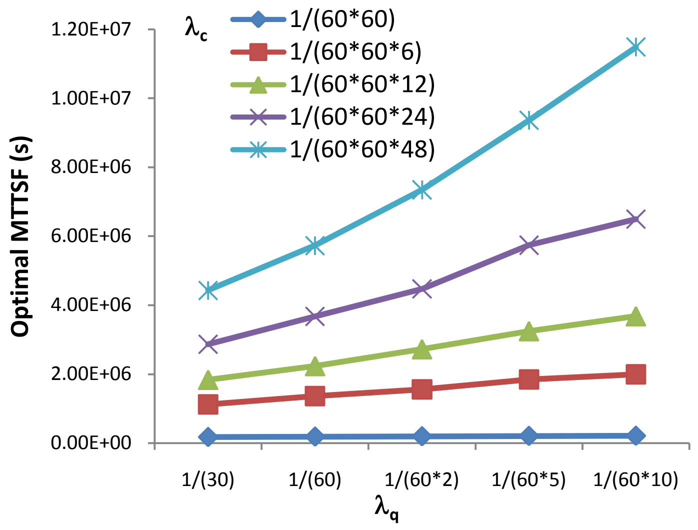


Figure 5- 4: Effect of λ_q on Optimal *MTTSF* with respect to λ_c with *m* = 5.

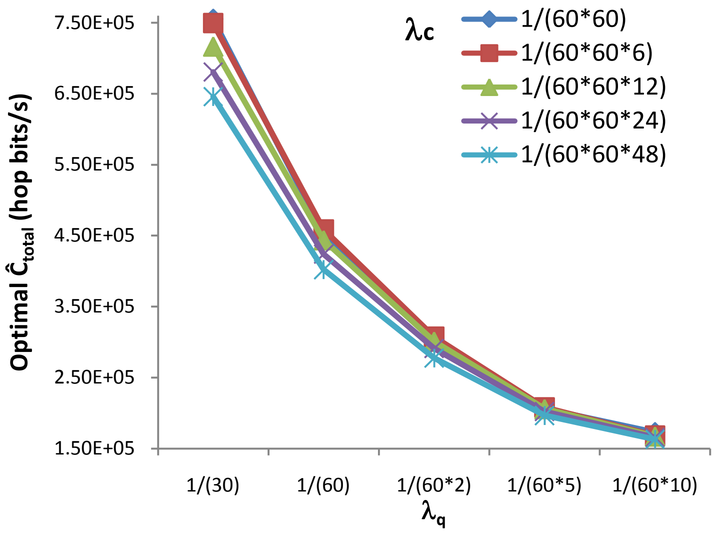


Figure 5- 5: Effect of λ_q on Optimal $\hat{C}_{total}$ with respect to λ_c with *m* = 5.

Next we analyze the general trend of optimal *MTTSF* or overall communication cost ($\hat{C}_{total}$) with respect to the base compromising rate (λ_c) and group communication rate (λ_q). Figure 5-4 shows optimal *MTTSF* identified as the group communication rate (λ_q) and the base node compromising rate (λ_c) vary over a broad range of parameter values with *m* fixed at 5 to isolate out its effect. We see that *MTTSF* increases as the group communication rate (λ_q) decreases. The reason is that as the group communication rate decreases, it lowers the probability of security failures due to C1 being satisfied. We also see that the effect of λ_q on *MTTSF* is especially pronounced when the base compromising rate (λ_c) is low. The reason is that when the base compromising rate is too high, security failures are mostly due to C2 being satisfied and the effect of λ_c dominates the effect of λ_q. Also not shown here, we have observed that as either λ_q or λ_c increases, the optimal T_{IDS} for maximizing *MTTSF* decreases in order to reduce the security vulnerability period for compromised nodes to perform attacks. Figure 5-5 correspondingly shows optimal $\hat{C}_{total}$ versus λ_q and λ_c. As expected as λ_q or λ_c increases, a higher $\hat{C}_{total}$ is generated. We also observe that (not shown in the figure) as λ_q or λ_c increases, the optimal T_{IDS} for minimizing $\hat{C}_{total}$ decreases. The reason is that to offset the increased traffic introduced, the system could perform IDS more often to evict nodes diagnosed as compromised nodes so as to reduce traffic due to group communication activities. The optimal T_{IDS} for minimizing $\hat{C}_{total}$ is dictated by increased IDS traffic versus reduced group communication traffic.

Next we analyze the effect of detection functions *D* (m_d) on *MTTSF*. Also as an example of applicability, we investigate how one can select the best detection interval (T_{IDS}) and detection function *D* (m_d) to optimize *MTTSF* while satisfying the performance requirement in terms of communication overhead, when given the attacker function *A* (m_c) detected at runtime.

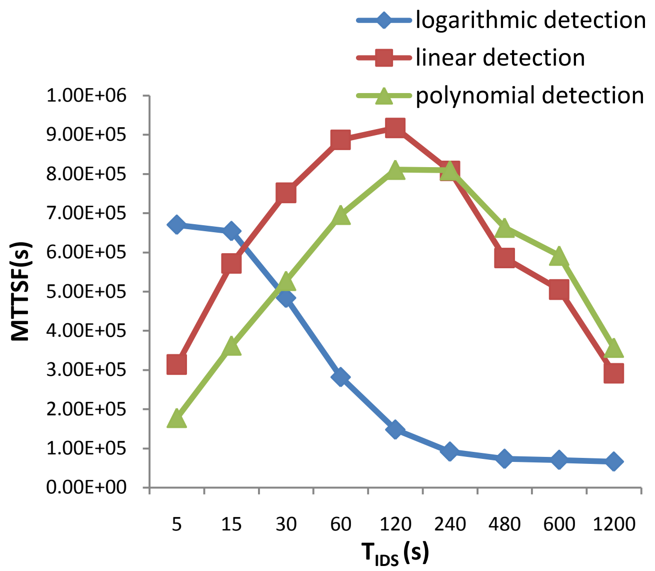


Figure 5- 6: Effect of T_{IDS} on *MTTSF* with respect to *D*(m_d) under linear time attacker function when m = 5.

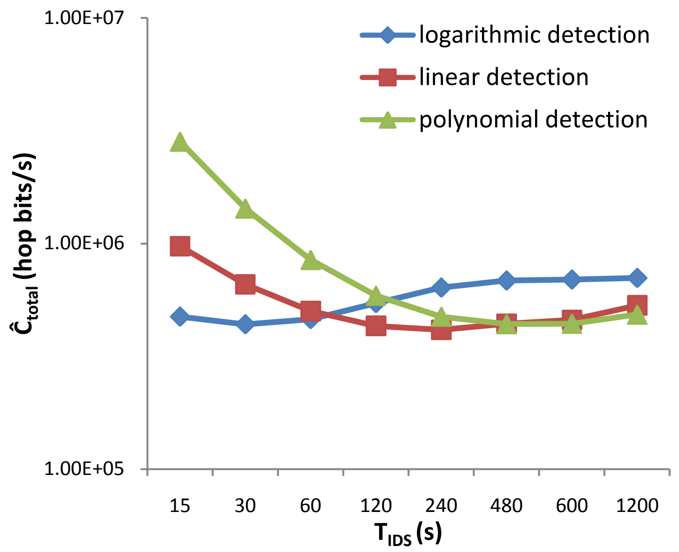


Figure 5- 7: Effect of T_{IDS} on $\hat{C}_{total}$ with respect to $D(m_d)$ under linear time attacker function when m = 5.

In Figure 5-6, we show *MTTSF* versus T_{IDS} for the three detection functions $D(m_d)$ given that the attacker function is linear. We see that each curve again has its own optimal T_{IDS}. The linear detection function $D_{linear}(m_d)$ shows the best performance at T_{IDS} = 120 s generating the highest *MTTSF* overall, while the logarithmic detection function $D_{log}(m_d)$ is the worst, particularly when T_{IDS} is sufficiently small. This tradeoff is attributed to the speed of detection (log, linear, or exponential) versus the speed of attack (linear). If the former is greater than the latter, many false positives may be generated; conversely, many compromised nodes may remain in the system. The linear detection function matches up with the linear attacker function the best among the three detection functions in terms of the tradeoff of the two ends. With similar reasoning, we see that the strongest polynomial detection function $D_{poly}(m_d)$ performs the best for a large T_{IDS} (e.g., T_{IDS} > 240 s) while the weakest logarithmic detection function $D_{log}(m_d)$ performs the best for a small T_{IDS} (e.g., T_{IDS} < 15 s).

Corresponding Figure 5-7 shows the overall communication cost ($\hat{C}_{total}$) versus T_{IDS} for the three detection functions $D(m_d)$ given that the attacker function is linear. Each curve in Figure 5-7 also has an optimal T_{IDS} that minimizes $\hat{C}_{total}$. The general trend of the optimal T_{IDS} identified is similar to that shown in Figure 5-6 although the exact optimal T_{IDS} points are different. The best performance of $\hat{C}_{total}$ is observed with linear detection at T_{IDS} = 240 *s* while the worst performance of $\hat{C}_{total}$ is shown with logarithmic detection under the ranges of T_{IDS} > 120 *s* and with polynomial detection under the ranges of $T_{IDS} \leq 120$ *s*, resulting in the best performance with linear detection overall. Also in terms of the optimal T_{IDS} identified to minimize $\hat{C}_{total}$, we see that a shorter optimal T_{IDS} is preferred with less aggressive logarithmic detection, since a shorter T_{IDS} contributes to nodes being evicted more often, consequently leading to less group communication activities. On the other hand, as the detection function becomes aggressive, i.e., polynomial detection, a longer optimal T_{IDS} is favorable to minimize $\hat{C}_{total}$ in order not to increase too much IDS related traffic more than needed due to aggressive IDS.

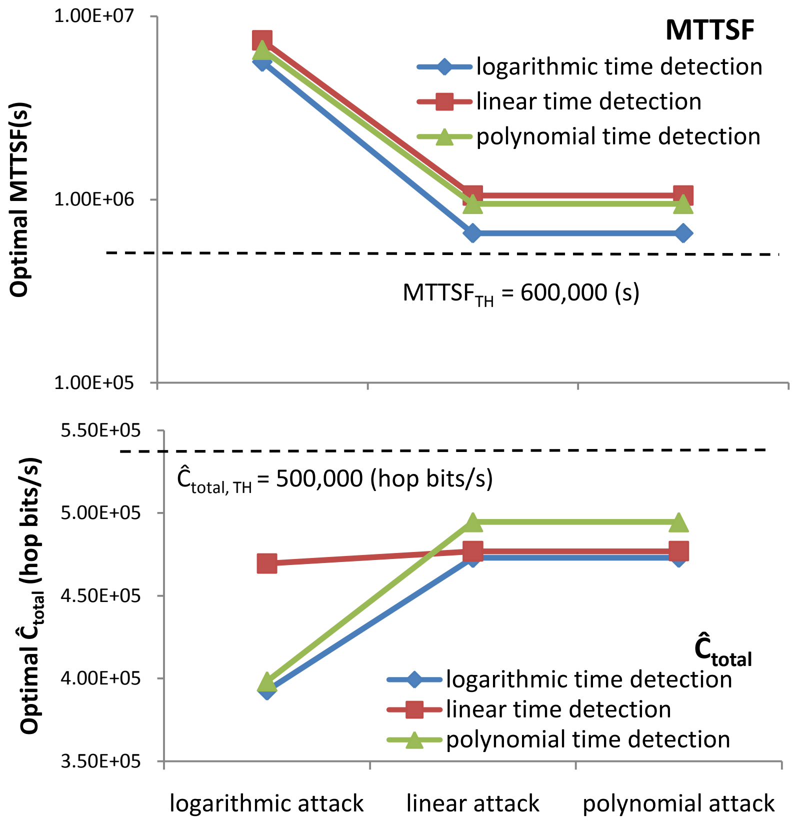


Figure 5- 8: Effect of Detection Functions on *MTTSF* and $\hat{C}_{total}$.

Table 5- 3: Optimal Settings for Generating *MTTSF* and $\hat{C}_{total}$ Corresponding to Figure 5-8.

$MTTSF_{TH}$ = 600,000 (s) $\hat{C}_{total,TH}$ = 500,000 (hop bits/s)	logarithmic detection		linear detection		polynomial detection	
	(m, T_{IDS})	*MTTSF* / $\hat{C}_{total}$	(m, T_{IDS})	*MTTSF* / $\hat{C}_{total}$	(m, T_{IDS})	*MTTSF* / $\hat{C}_{total}$
logarithmic attack	(9, 15)	5,637,970	(9, 60)	7,372,577	(9, 240)	6,565,735
		392,698		469,471		398,400
linear attack	(7, 15)	653,848	(7, 120)	1,053,356	(7, 240)	948,253
		472,934		476,857		494,642
polynomial attack	(5, 15)	653,671	(7, 120)	1,053,146	(7, 240)	948,048
		473,018		476,932		494,582

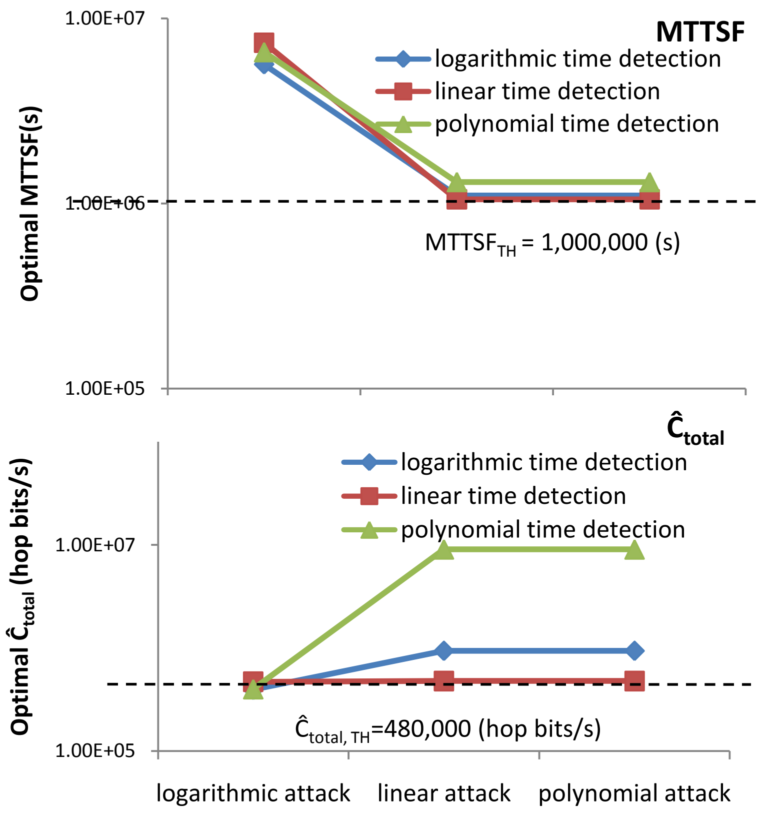


Figure 5- 9: Effect of Detection Functions on *MTTSF* and $\hat{C}_{total}$ under Stringent Constraints.

Table 5- 4: Optimal Settings for Generating *MTTSF* and $\hat{C}_{total}$ Corresponding to Figure 5-9.

$MTTSF_{TH}$ = 1,000,000 (s) $\hat{C}_{total,TH}$ = 480,000 (hop bits/s)	logarithmic detection		linear detection		polynomial detection	
	(m, T_{IDS})	*MTTSF* / $\hat{C}_{total}$	(m, T_{IDS})	*MTTSF* / $\hat{C}_{total}$	(m, T_{IDS})	*MTTSF* / $\hat{C}_{total}$
logarithmic attack	(9, 15)	5,637,970	(9, 60)	7,372,577	(9, 240)	6,565,735
		392,698		469,471		398,400
linear attack	(9, 5)	1,104,114	(7, 120)	1,053,356	(9, 5)	1,304,663
		938,511		476,857		9,041,128
polynomial attack	(9, 5)	1,103,882	(7, 120)	1,053,146	(9, 5)	1,304,404
		938,684		476,932		9,033,611

Below we exemplify the selection of optimal design settings in terms of the base intrusion detection interval (T_{IDS}), the number of vote-participants (m), and the IDS detection function $D(m_d)$, when given an attacker function detected at runtime and performance constraints set by the GCS. Figure 5-8 shows the optimal *MTTSF* and the associated $\hat{C}_{total}$ values obtainable under three detection functions (logarithmic, linear and exponential), when given an attacker function (in the X coordinate) and the performance constraints set by the GCS system, i.e., $MTTSF_{TH}$ =

600,000 *s* and $\hat{C}_{total,TH}$ = 500,000 *hop bits/s*. Table 5-3 lists the actual optimal values obtained as well as the optimal settings (m, T_{IDS}) under which the optimal values are obtained. Suppose due to the criticality of mission-oriented applications in MANETs, the design goal is to maximize *MTTSF* while satisfying performance constraints. One can then do a table lookup to select the linear IDS detection function to achieve the goal given an attack function detected at runtime. Specifically, from Table 5-3 one would select to apply the settings m = 9 and T_{IDS} = 60 *s* for logarithmic attack, and m = 7 and T_{IDS} = 120 *s* for both linear and polynomial attacks. As discussed earlier, this table lookup can be performed upon detection of the attacker function from observing historical data on compromised nodes that have been detected by IDS.

Figure 5-8 is for the scenario in which all three detection functions satisfy the imposed performance constraints. Figure 5-9 shows the optimal *MTTSF* and the associated $\hat{C}_{total}$ values obtainable for a scenario in which performance constraints imposed are more stringent with $MTTSF_{TH}$ = 1,000,000 *s* and $\hat{C}_{total,TH}$ = 480,000 *hop bits/s*. Table 5-4 lists the actual optimal values obtained as well as the optimal settings (m, T_{IDS}) under which the optimal values are obtained. We see that only linear detection is able to meet the performance constraints when the attacker function is linear or polynomial despite logarithm detection and exponential detection can generate a higher *MTTSF*. Consequently, the optimal settings identified is m = 7 and T_{IDS} = 120 *s* with the linear detection function being to maximize *MTTSF* while satisfying the $\hat{C}_{total,TH}$ requirement. When the attacker function is logarithmic, we see that all three detection functions are able to satisfy the imposed performance constraints. In this case, since the linear detection function with m = 9 and T_{IDS} = 60 *s* generates the highest *MTTSF*, it should be selected for achieving higher survivability for the mission-oriented GCS in MANET environments.

5.3.2 Sensitivity Analysis

In this section, we investigate the sensitivity of *MTTSF* versus T_{IDS} with respect to a number of key parameters, including the system failure definition (SF1 versus SF2), the number of vote-participants selected for performing majority voting in voting-based IDS (m), the group communication rate (λ_q) and the base compromising rate (λ_c). We also perform sensitivity analysis in both single-hop and multi-hop MANET environments.

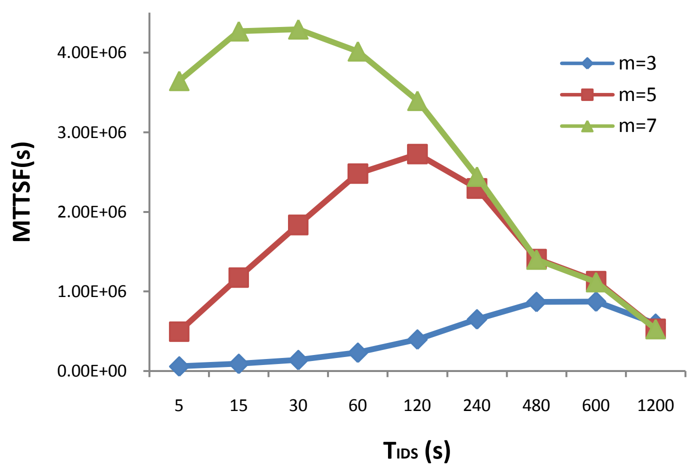


Figure 5- 10: Effect of T_{IDS} on *MTTSF* under varying *m* in single-hop MANETs.

Figure 5-10 shows the effect of intrusion detection interval (T_{IDS}) on *MTTSF* as the number of vote-participants (*m*) in voting-based IDS changes in single-hop MANETs in which only one mobile group exists in the GCS during the lifetime. We see that there exists an optimal T_{IDS} that maximizes *MTTSF*. As T_{IDS} increases, *MTTSF* increases until its optimal point reaches, and then *MTTSF* decreases after the optimal point. The reason of having increasing *MTTSF* as T_{IDS} increases initially is that triggering IDS too often has the effect of evicting nodes quickly in the system due to false positives, thus resulting in a quick system failure because of C2. Here we note that negative effects of IDS are mostly due to false positives (diagnosing good nodes as bad nodes) and the effects are more pronounced when IDS is triggered more often. The reason of having decreasing *MTTSF* as T_{IDS} increases further past the optimal point is that when IDS is not been triggering often enough, more compromised nodes will remain in the system, thus resulting in system failures mostly due to C1 and just partly due to C2.

We also see from Figure 5-10 the effect of *m* (the number of vote-participants in voting-based IDS) on *MTTSF*. When *m* is large, the false alarm probability (P_{fp} + P_{fn}) is low because more nodes will participate in the voting process, thus reducing the possibility of collusion by compromised nodes. Consequently, when *m* is large, we observe a high *MTTSF*. Conversely, when *m* is small, the false alarm probability is relatively large, resulting in a small *MTTSF*. This trend is generally true when the mobile user population is sufficiently high so that the probability of being able to find *m* nodes is sufficiently high. Lastly we observe that a smaller *m* results in a large T_{IDS} being used to maximize *MTTSF* to offset the adverse effects of IDS with large false positives.

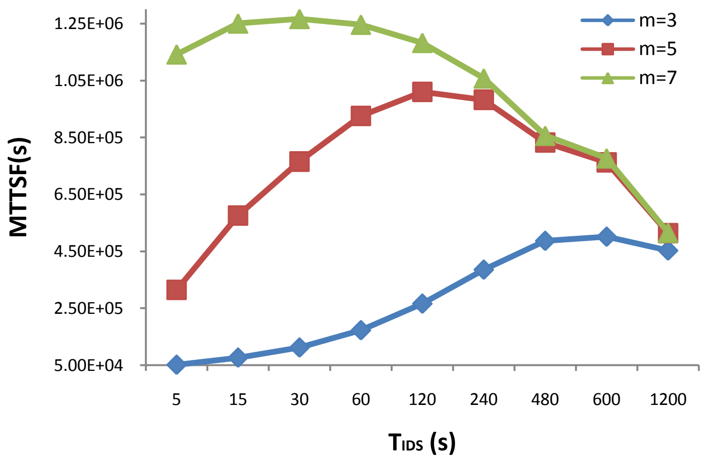


Figure 5- 11: Effect of *T_{IDS}* on *MTTSF* under varying *m* in multi-hop MANETs based on SF1.

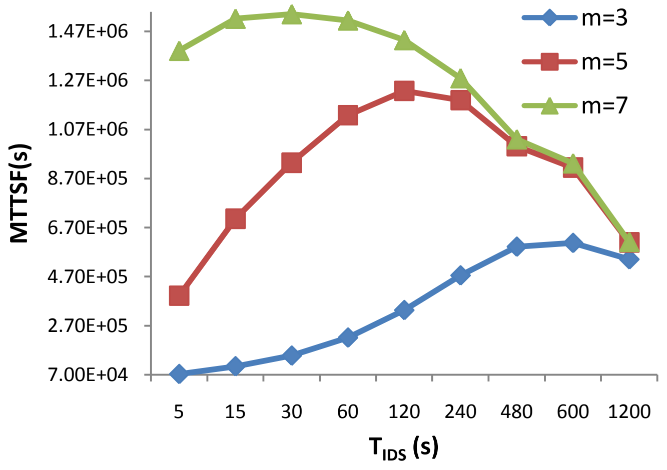


Figure 5- 12: Effect of *T_{IDS}* on *MTTSF* under varying *m* in multi-hop MANETs based on SF2.

Figures 5-11 and 5-12 show the effect of intrusion detection interval (T_{IDS}) on *MTTSF* as the number of vote-participants (*m*) varies in multi-hop MANETs for system failure definitions SF1 and SF2, respectively. Here nodes are connected by multiple hops so that multiple groups exist in the system due to occurrences of group merge/partition events in the GCS. Similar to Figure 5-10, we see from Figures 5-11 and 5-12 that an optimal intrusion detection interval (T_{IDS}) exists to maximize *MTTSF*. Further, the optimal T_{IDS} value increases as *m* decreases. The same reasoning used for explaining these trends in Figure 5-10 applies. We observe that *MTTSF* of the GCS in single-hop MANETs is comparatively higher than *MTTSF* of the same GCS in multi-hop MANETs under either system failure definition (SF1 or SF2). The reason is that when there are multiple groups in the system, the node density in each group tends to be small, given that the GCS is initially deployed with N_{init} = 150 users. Here we see the adverse effect of breaking the system into multiple mobile groups on *MTTSF*. Lastly, we observe from Figures 5-11 and 5-12 that *MTTSF* of the system under SF2 is much higher than that of the system under SF1 because SF2 allows the mission to continue as long as one mobile group exists.

Below we test the sensitivity of the results with respect to the group communication rate (λ_q) and the base node compromising rate (λ_c). Figure 5-13 shows the sensitivity of *MTTSF* with respect to the group communication rate (λ_q) in single-hop MANETs. We observe that when λ_q is low so the data-leak attack is not performed often, the positive effect of IDS is pronounced, leading to a high *MTTSF*. On the other hand, when λ_q is high so the data-leak attack is frequent, the negative effect of IDS is pronounced, so *MTTSF* is low. We also observe that the optimal T_{IDS} becomes smaller as λ_q increases because the system prefers removing compromised nodes as soon as possible so that compromised nodes would not have a chance to perform data-leak attacks. Another observation is that when T_{IDS} is sufficiently small, e.g., T_{IDS} < 60 *s*, *MTTSF* remains about the same regardless of the magnitude of λ_q. This is because when IDS is being invoked too frequently, the adverse effect of false positives dominates the positive effect of IDS.

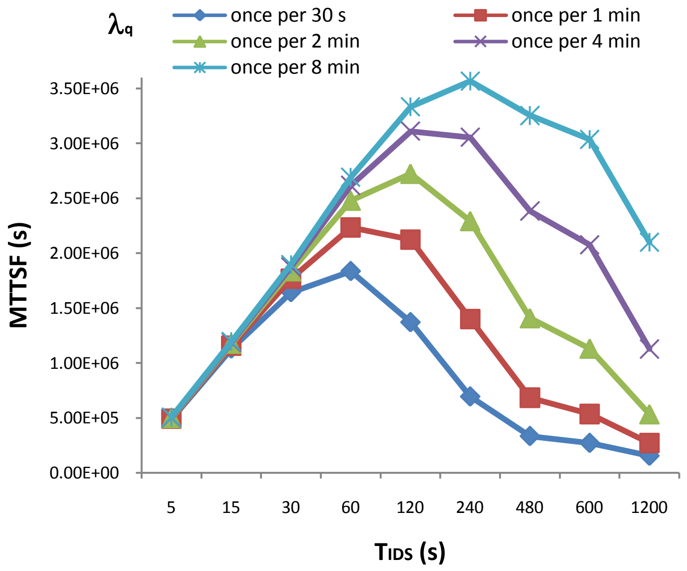


Figure 5- 13: Sensitivity of *MTTSF* with respect to λ_q in Single-hop MANETs.

Figures 5-14 and 5-15 test the sensitivity of *MTTSF* with respect to the group communication rate (λ_q) in multi-hop MANETs based on SF1 and SF2, respectively. We see that there exists optimal T_{IDS} under which *MTTSF* is maximized and that the optimal point decreases as λ_q increases, exhibiting the same trend as in single-hop MANETs. Comparing single-hop MANETs versus multi-hop MANETs, however, we observe the optimal T_{IDS} is smaller in single-hop MANETs under identical conditions. The reason is that single-hop MANETs tend to have more group members because all members are within one-hop radio range. Consequently, single-hop MANETs need to perform IDS more frequently to prevent potentially more compromised nodes from attacking the system causing C1 or C2 to be violated. Comparing *MTTSF* in multi-hop MANETs based on SF1 and SF2, we observe that a higher *MTTSF* is obtained under SF2 because the system fails when all groups fail as opposed to when one group fails.

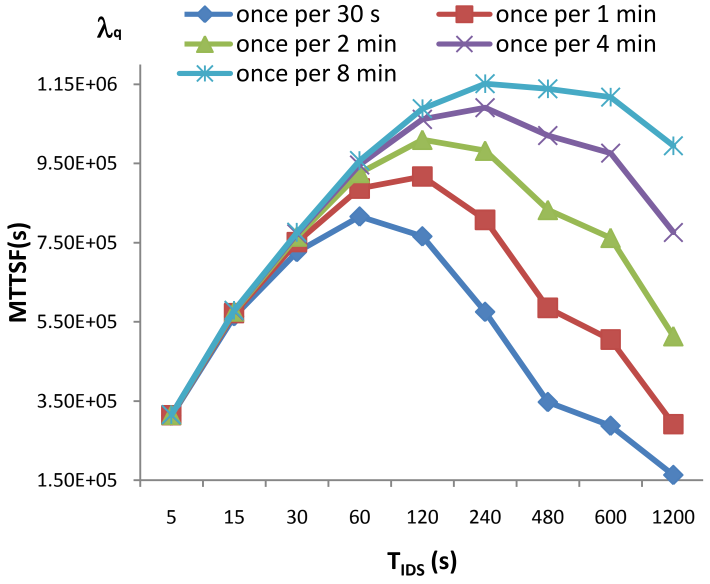


Figure 5- 14: Sensitivity of *MTTSF* with respect to λ_q in Multi-hop MANETs based on SF1.

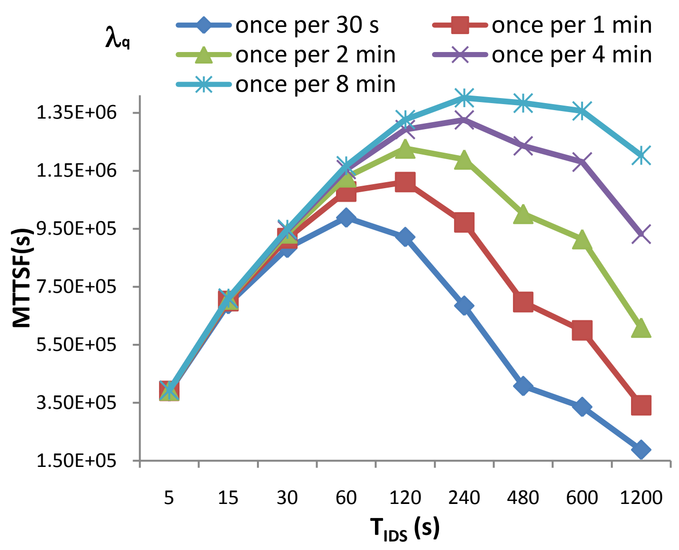


Figure 5- 15: Sensitivity of *MTTSF* with respect to λ_q in Multi-hop MANETs based on SF2.

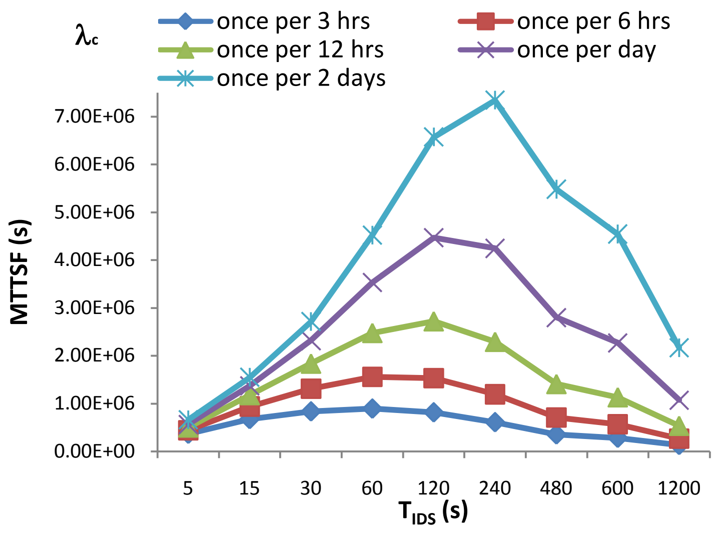


Figure 5- 16: Sensitivity of *MTTSF* with respect to λ_c in Single-hop MANETs.

Next we test the sensitivity of *MTTSF* with respect to the base compromising rate (λ_c) in single-hop MANETs. Figure 5-16 summarizes the results. We first observe that as λ_c increases, *MTTSF* decreases because a higher λ_c will cause more compromised nodes to be present in the system. We also observe that the optimal T_{IDS} decreases as λ_c increases. This is because when more compromised nodes exist, the system needs to execute IDS more frequently to maximize *MTTSF*. Finally, we observe that when λ_c is low, the effect of T_{IDS} on *MTTSF* is especially pronounced. Thus, IDS is more effective when λ_c is sufficiently low.

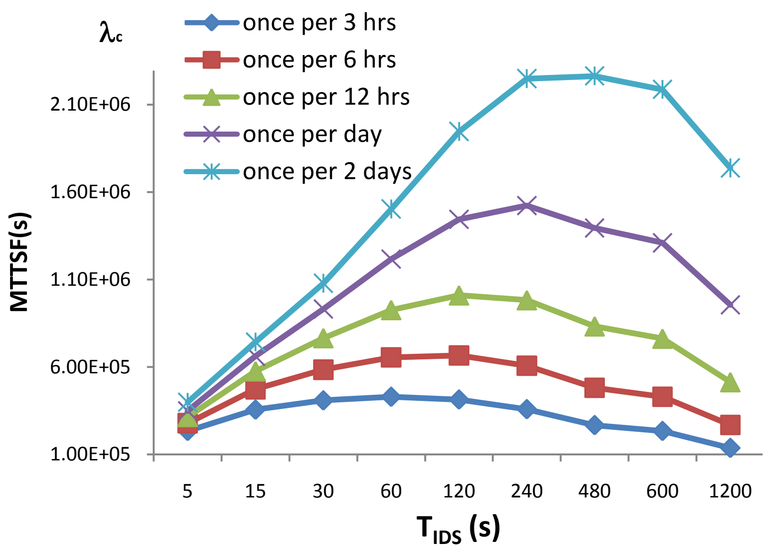


Figure 5- 17: Sensitivity of *MTTSF* with respect to λ_c in Multi-hop MANETs based on SF1.

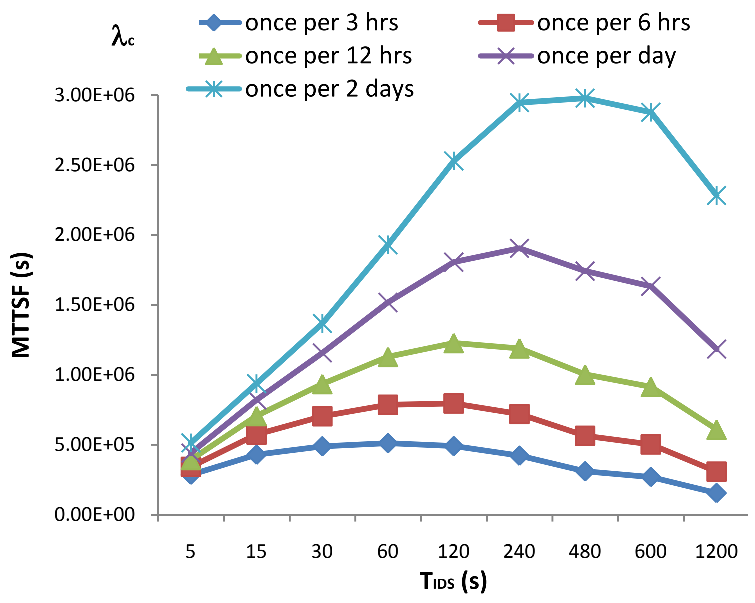


Figure 5- 18: Sensitivity of *MTTSF* with respect to λ_c in Multi-hop MANETs based on SF2.

Correspondingly Figures 5-17 and 5-18 summarize the sensitivity of *MTTSF* with respect to the compromising rate (λ_c) in multi-hop MANETs based on SF1 and SF2, respectively. The sensitivity result exhibited in Figures 5-17 and 5-18 for multi-hop MANETs is remarkably similar in trend to that in Figure 5-16 for single-hop MANETs. Comparing single-hop MANETs versus multi-hop MANETs, we observe two results: (a) single-hop MANETs have higher *MTTSF* because more members exist in single-hop MANETs, and (b) the optimal T_{IDS} is smaller in single-hop MANETs under identical conditions because the system tends to execute IDS more frequently when there are more members in a group. Comparing multi-hop MANETs under SF1 and SF2, we again observe a significantly higher *MTTSF* being obtained under SF2 due to the less stringent system failure definition for the GCS mission being executed.

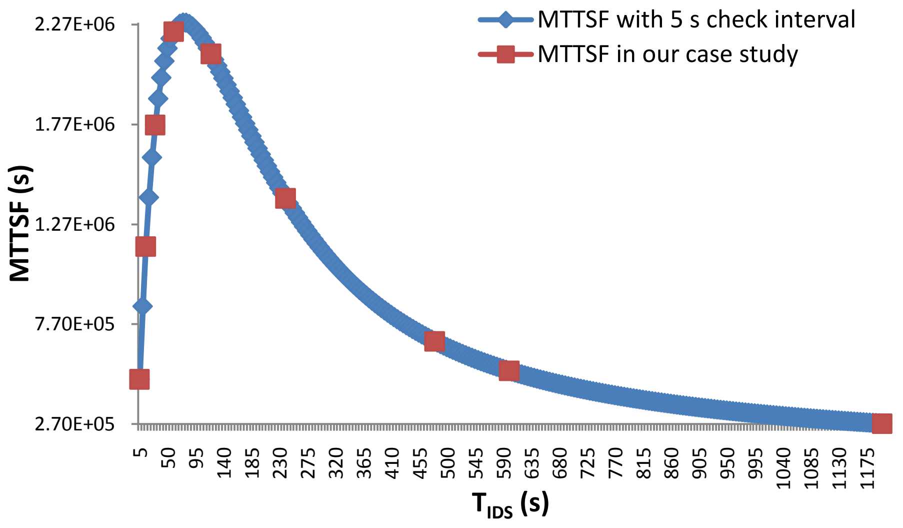


Figure 5- 19: Optimality Test for IDS.

In all the results presented so far, we have claimed the intrusion detection interval (T_{IDS}) identified for maximizing *MTTSF* as the "optimal" T_{IDS}. We support the claim of "optimization" by (1) identifying a maximal range of parameter values for each parameter; (2) testing the sensitivity of the optimal setting identified with respect to the data-point granularity over the maximal range identified. As an example, Figure 5-19 shows a curve-fit *MTTSF* function formed with a very fine data-point granularity, verifying that the optimal detection interval identified in voting-based IDS is indeed optimal over its maximal range of values.

5.4 Simulation Validation

We have conducted a simulation study to validate analytical results. The simulation program is implemented based on a discrete-event simulation language called *SMPL* [81]. The default parameter values are shown in Table 5-5.

We explain the default parameter values used in the simulation as follows. The wireless bandwidth is considered limited and set at 1Mbps. The false negative probability and the false positive probability of host-based IDS are set at 1% each, reflecting the presence of a medium to high quality host-based IDS. The ratio of join to leave events is set to 4, reflecting the fact that nodes join a group much faster than they leave a group. Group members communicate to other group members once per 2 minutes. The rate at which nodes are compromised is once per 12 hours, reflecting a medium-high level of attack strength by the attackers. We change the values of key parameters to analyze their effects on the simulation results.

Table 5- 5: Main Parameters and Default Values for Simulation Validation.

Parameter	Value	Parameter	Value	Parameter	Value
λ	1/(60*60)	σ	1/(60*60*32)	T_{RB}	5 *s*
μ	1/(60*60*4)	BW	1Mbps	T_{LB}	2 *s*
T_{IDS}	5 – 1200 *s*	N_{init}	150 nodes	m	3
T_{status}	300 *s*	$D(m_d)/A(m_c)$	Linear	R	200 *m*
λ_c	1/(60*60*12)	r	500 *m*	b_{vote}	100 *bits*
λ_q	1/(60*2)	b_{GDH}	64 *bits*	$b_{m\text{-}list}$	100 *bits*
$p1$	1 %	b_{GC}	800 *bits*	M_{alive}	32 *bits*
$p2$	1 %	b_s	400 *bits*	U_{view}	500 *bits*

We populate the MANET area based on a selected node population. For example, we randomly place 150 nodes within the operational area with size $(500)^2\ \pi\ km^2$ in our simulation. In this case, the initial number of member nodes, i.e., $N\ \lambda/(\lambda+\mu)$, is 120 approximately and they are scattered in the operational area. All nodes can be connected through multiple hops using a per-hop wireless radio range = 200 *m*. Multiple groups may be observed in the operational area.

Each node in its lifecycle could generate six events, namely, GROUP JOIN, GROUP LEAVE, BEACON, GROUP COMMUNICATION, GROUP MERGE, GROUP PARTITION, INTRUSION DETECTION, and COMPROMISE. GROUP JOIN and GROUP LEAVE events occur with rates of λ and μ respectively. We assume the inter-arrival time is exponentially distributed. Upon the occurrence of a GROUP PARTITION or GROUP MERGE event, the group view and the associated membership changes are updated. The time a GROUP PARTITION event or a GROUP MERGE event occurs depends on the node distribution and user mobility. We check occurrences of group merge/partition events by a timer event. The GROUP COMMUNICATION and BEACON events are scheduled periodically with a fixed interval. The GROUP COMMUNICATION event occurs with rate λ_q. If a compromised node triggers GROUP COMMUNICATION event, we consider the system as having experienced a security failure due to violation of Condition C1. To expedite data collection, we also turn off the host-IDS capability, that is, not detecting if the sender is suspicious of compromised, so that whenever a compromised node involves in GROUP COMMUNICATION event the system fails. The INTRUSION DETECTION event occurs periodically at a rate given by the linear attacker function in Equation 5-5. When an INTRUSION DETECTION event occurs, each good node is tested with the false positive probability to see if it has been diagnosed by voting-based IDS as a bad node and each bad node is tested with the false negative probability to see if has been diagnosed by IDS as a bad node. Lastly, the COMPROMISE event occurs at a rate given by the linear attacker function in Equation 5-4.

While we can consider any mobility model in the simulation, we adopt the *random waypoint mobility model* [13] because of its popularity to model the movement of a node with the mobility rate being σ, the pause time being 0, and the speed being:

$$S(\sigma) = \frac{2r}{expntl(1/\sigma)} \tag{5-27}$$

where r is the radius of the area and $expntl(1/\sigma)$ returns a random number exponentially distributed with rate σ.

In modeling the COMPROMISE event, a good node is selected to-be-compromised (TBC) when we schedule a COMPROMISE event. A TBC node, like a compromised but undetected node, can freely join or leave a group. A TBC node then is compromised when the COMPROMISE event occurs, even if it already leaves the group it originally belongs to. If the TBC node is falsely identified as a bad node by IDS before the COMPROMISE event occurs, the COMPROMISE event is dropped and a new COMPROMISE event is scheduled right after the TBC node is evicted out of the system due to false positives of IDS. Upon a GROUP MERGE or GROUP PARTITION event, all previously scheduled COMPROMISE events in all groups involved in the group merge/partition event are canceled and a new COMPROMISE event is scheduled in each involved group.

A simulation run ends whenever a system failure occurs, either due to violation of Condition C1 or Condition C2. We collect the system lifetime obtained when the system failure occurs and execute another run from scratch. Figure 5-20 compares simulation results obtained versus analytical results for *MTTSF* versus T_{IDS}. The simulation results displayed are the average values out of 100 simulation runs. We see that the simulation results exhibit a similar trend compared with analytical results, identifying the optimal T_{IDS} at 60 as analytical results indicate. The mean percentage difference (MPD) between analytical results and simulation results is 10% with the standard error (SE) being 5868. The MPD and SE are defined by [117]:

$$MPD = \frac{\sum_{i=1}^{n} \frac{|x_i - y_i|}{y_i}}{n}, \qquad SE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - y_i)^2}{n}} \tag{5-28}$$

where x_i is a simulation result value, y_i is an analytical result value, and *n* is the number of result points. The SE value of 5868 is approximately 5.7% out of 102294, the optimal *MTTSF* at the optimal T_{IDS} identified at 60. Since the MPD and SE values are sufficiently small, we conclude that simulation results obtained match well with analytical results.

The reason of having a slight difference between analytical and simulation results may be attributed to the fact that in the analytical model we consider equal-size grouping, while in simulation groups do not necessarily have the same size. Consequently, the rate at which system failures are triggered (due to group failures based on SF1) may also be different. Nevertheless, overall we see a good correlation of simulation results versus analytical results especially in the overall trend predicted and we conclude that the analytical results obtained are valid.

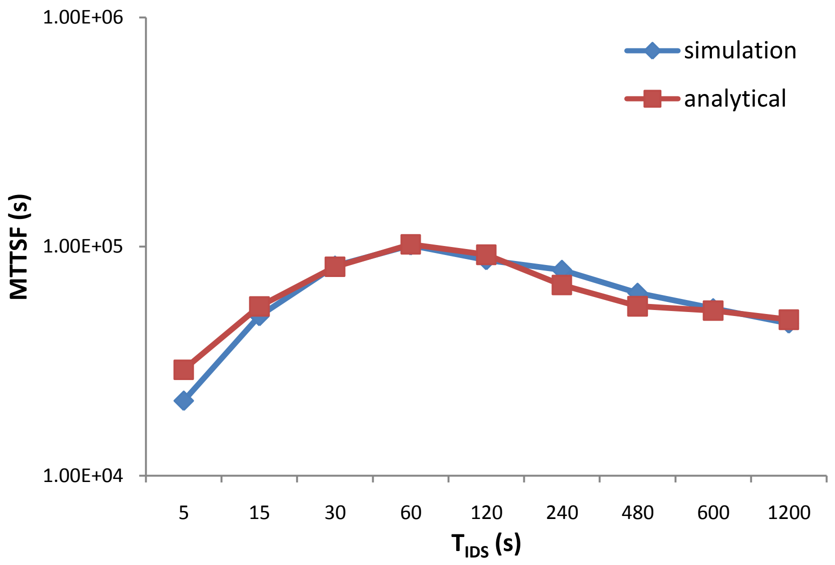


Figure 5- 20: Analytical versus Simulation Results – Effect of T_{IDS} on *MTTSF*.

5.5 Summary

In this chapter, we have proposed and analyzed voting-based IDS against inside attackers for secure GCSs in MANETs. The intrusion detection interval (T_{IDS}) used by voting-based IDS can be adjusted based on the behavior of inside attackers. Our analysis revealed the intrinsic tradeoff between security (measured by the *MTTSF* metric) and performance (measured by the overall communication cost $\hat{C}_{\mathrm{total}}$ metric). When given a GCS characterized by a set of parameter values, we showed that there exists an optimal detection interval (T_{IDS}) that maximizes *MTTSF* as well as satisfying the constraint on the communication traffic ($\hat{C}_{total}$). The existence of the optimal detection interval (T_{IDS}) for maximizing *MTTSF* is attributed to the tradeoff between the positive effect of IDS (i.e., identifying compromised nodes and evicting them properly to prolong system lifetime) versus the adverse effect of IDS (i.e., false negatives and false positives generated by IDS). The existence of the optimal detection interval (T_{IDS}) for minimizing $\hat{C}_{total}$ is attributed to the tradeoff between the IDS communication traffic versus the group communication traffic. We have also performed sensitivity analysis of *MTTSF* versus T_{IDS} with respect to a number of key parameters, including the system failure definition (SF1 versus SF2), the number of vote-participants selected for performing majority voting in voting-based IDS (*m*), the group communication rate ($\lambda_{\boldsymbol{q}}$) and the base compromising rate ($\lambda_{\boldsymbol{c}}$) in single-hop as well as multi-hop MANET environments.

We have investigated three ways to perform IDS detection and how the system could adjust the IDS detection level in response to the attacker strength detected at runtime in order to maximize *MTTSF* and minimize $\hat{C}_{total}$ dynamically. We discovered that we could select the best detection function (*logarithmic*, *linear*, or *polynomial*) in response to the attacker strength to maximize *MTTSF* without experiencing much of the adverse effect of IDS. The results obtained in terms of *MTTSF* and $\hat{C}_{total}$ versus T_{IDS} allow the system designer to select the best intrusion detection interval (T_{IDS}) to maximize *MTTSF*, or minimize $\hat{C}_{total}$, depending on the security versus performance requirements, or to maximize MTTSF while satisfying the $\hat{C}_{total}$ performance requirements. To apply the results, one can cover a wide range of values of model parameters and build a table at static time listing the selection of the intrusion detection interval (T_{IDS}) that can both maximize MTTSF and/or minimize the overall communication cost ($\hat{C}_{total}$). Then, at runtime, the system can perform a table lookup operation to select the best intrusion detection function, the best IDS detection interval (T_{IDS}) and the best number of vote-participants for voting-based IDS based on statistical information collected dynamically. QoS-aware IDS protocols developed in this chapter can be combined with QoS-aware key management protocols to deal with both insider attacks and outsider attacks. In the dissertation research, we will investigate the integration of QoS-aware IDS protocols with threshold-based periodic batch rekeying, which is the subject of Chapter 6. We will also investigate the integration of QoS-aware IDS protocols with region-based group key management for scalability and dynamic reconfigurability, which is the subject of Chapter 8.

Chapter 6 INTRUSION DETECTION INTEGRATED WITH THRESHOLD-BASED PERIODIC BATCH REKEYING

In this chapter, we integrate QoS-aware IDS with threshold-based periodic batch rekeying for GCSs in MANETs where nodes communicate through multi-hops. Our approach is unique in that there is no prior work evaluating the tradeoff of performance and security properties of GCSs equipped with QoS-aware IDS to deal with insider attacks while being integrated with threshold-based periodic batch rekeying to prevent outsider attacks in MANET environments. The goal is to demonstrate optimal settings including the best intrusion detection interval and the best batch rekey interval under which the system lifetime in terms of *MTTSF* is maximized while satisfying performance requirements. The content is largely based on our published conference paper [23] and a submitted journal paper [29].

Section 6.1 briefly describes how QoS-aware intrusion detection and threshold-based periodic batch rekeying protocols may be integrated to result in secure QoS-aware GCSs in MANETs. Section 6.2 develops a mathematical model based on SPN for performance evaluation. It also describes how model parameters are parameterized. Section 6.3 presents numerical results identifying optimal settings in terms of optimal rekeying thresholds and intrusion detection intervals to maximize *MTTSF* while satisfying performance requirements. The effectiveness of the integration design is demonstrated by performing a comparative analysis with a baseline system with individual rekeying. Section 6.4 concludes this chapter.

6.1 Integration of QoS-Aware IDS with Threshold-based Batch Rekeying

Our proposed GCS is equipped with QoS-aware IDS protocols (Chapter 5) to deal with inside attackers and threshold-based periodic batch rekeying protocols (Chapter 4) to deal with outside attackers in multi-hop MANETs.

For the selection of m vote-participants in voting-based IDS, each node periodically exchanges its routing information, location, and id with its neighboring nodes. With respect to a target node, nodes that are $H_{nb}(m)$-hop away are candidates as vote-participants where m is a design parameter. A node with the smallest id (or the largest id as a tie breaker) will elect itself as the coordinator, select m nodes randomly (including itself), and broadcast this list of m selected vote-participants to all group members. After m vote-participants for a target node are selected this way, each vote-participant independently votes for or against the target node by disseminating its vote to all group members. Vote authenticity is achieved via preloaded public/private key pairs. All group members know who m vote-participants are, and, based on votes received, can determine whether or not a target node is to be evicted. Under batch rekeying, all evicted nodes along with newly join and leave nodes will be processed at the beginning of the next batch rekey interval and a new group key will be generated based on CKA among current group members.

We consider three rekeying protocols for GCSs in multi-hop MANETs:

- ***Individual rekeying with CKA***: A rekeying operation based on *CKA* is performed right after every join/leave/eviction request.
- ***Trusted and Untrusted Double Threshold-based rekeying with CKA (TAUDT-C)***: A rekeying is performed after a threshold (*k1*, *k2*) is reached, where *k1* is the number of requests from trusted nodes (i.e., trusted join nodes plus trusted leave nodes) and *k2* is the number of requests due to evictions for the nodes detected by IDS as compromised in the system. That is, when either *k1* or *k2* is reached, a rekeying operation based on *CKA* is performed. This protocol extends *TAUDT* in Chapter 4.
- ***Join and Leave Double Threshold-based rekeying with CKA (JALDT-C)***: A rekeying is performed after a threshold (*k1*, *k2*) is reached, where *k1* is the number of requests from join nodes (i.e., trusted join nodes) and *k2* is the number requests from trusted leave nodes plus

forced evictions for the nodes detected by IDS as compromised in the system. This protocol extends *JALDT* in Chapter 4.

TAUDT-C and *JALDT-C* extend *TAUDT* and *JALDT* by utilizing *CKA* for distributed control and removing a single point of failure in MANETs. For brevity, we will just call them *TAUDT* and *JALDT* in this chapter. Without loss of generality, we again consider GDH.3 (called GDH for brevity) [80] as the *CKA* protocol for secret key generation. To describe behaviors of attackers and IDS, we use *linear time attacker* and *linear periodic detection* functions as default. Because of the use of threshold-based periodic batch rekeying for performance enhancement and QoS-awareness reasons, members diagnosed as compromised by IDS may stay in the system without being evicted immediately until a rekeying operation is performed. This introduces a security vulnerability period during which compromised, detected members may perform attacks using fake identities (e.g., impersonation attacks).

6.2 Performance Analysis

6.2.1 Performance Model

We use the ***MTTSF*** to measure security and the average ***Service Response Time ($\bar{R}$)*** to measure performance properties of the proposed GCS in MANETs.

We develop a mathematical model based on an SPN as shown in Figure 6-1 to describe the behaviors of a GCS instrumented with IDS to cope with insider attacks, and batch rekeying to deal with outsider attacks. Our goal is to identify optimal settings to maximize *MTTSF* while satisfying imposed performance requirements in terms of $\bar{R}$. Table 6-1 summarizes the model parameters used.

Table 6- 1: Model Parameters.

Symbol	Meaning
$\boldsymbol{A}$	Operational area $A = \pi r^2$ (unit: m^2)
$\boldsymbol{r}$	Radius of an operational area (m)
$\boldsymbol{H}$	Average number of hops between a sender and a receiver
$\boldsymbol{\lambda}$	Arrival rate of join requests (s^{-1})
$\boldsymbol{\mu}$	Arrival rate of leave requests (s^{-1})
$\boldsymbol{T_{IDS}}$	Initial intrusion detection interval (s)
$\boldsymbol{\lambda_c}$	Initial attacker rate (s^{-1})
$\boldsymbol{m_d}$	Degree of compromised nodes that have been detected by IDS
$\boldsymbol{D\ (m_d)}$	A linear detection function that dynamically returns a periodic detection rate based on m_d, i.e., $D\ (m_d) = m_d\ (1/T_{IDS})$ (unit: s^{-1})
$\boldsymbol{m_c}$	Degree of compromised nodes currently in the system
$\boldsymbol{A\ (m_c)}$	A linear attacker function based on m_c that dynamically returns the rate at which nodes are compromised, i.e., $A\ (m_c) = m_c\lambda_c$ (unit: s^{-1})
$\boldsymbol{H_{nb}(m)}$	A function that returns the hop number of neighboring nodes based on m
$\boldsymbol{\lambda_q}$	Group data communication rate per node (s^{-1})
$\boldsymbol{p1}$	False negative probability of host-based IDS
$\boldsymbol{p2}$	False positive probability of host-based IDS
$\boldsymbol{T_{cm}}$	Communication time for broadcasting a rekey message (s)
$\boldsymbol{b_{GDH}}$	Length of an intermediate value in applying GDH (*bits*)
$\boldsymbol{b_{GC}}$	Packet size for group communication activities (*bits*)
$\boldsymbol{m}$	Number of vote-participants against a target node
$\boldsymbol{BW}$	Wireless network bandwidth (*Mbps*)
$\boldsymbol{N_{init}}$	Initial number of member nodes in the system
$\boldsymbol{N}$	Number of current trusted member nodes
MTTSF	Mean time to security failure (s)
$\boldsymbol{\bar{R}}$	Average service response time per group communication operation (s)
$\boldsymbol{\Lambda_J}$	Aggregate group join rate (s^{-1})
$\boldsymbol{\Lambda_L}$	Aggregate group leave rate (s^{-1})
$\boldsymbol{T_{RTS}}$	Transmission delay for RTS (request-to-send) (s)
$\boldsymbol{T_{CTS}}$	Transmission delay for CTS (clear-to-send) (s)
SIFS	Short inter-frame space (s)
DIFS	Distributed inter-frame space (s)
$\boldsymbol{T_{slot}}$	Slot time in random backoff (s)
E[CW]	Average contention-window size (unit: slot)
$\boldsymbol{T_{com}}$	Transmission delay for a packet (*sec*)
$\boldsymbol{T_b}$	Wireless network delay including channel contention time (s)
$\boldsymbol{T_c}$	Channel contention delay with an idle channel (s)
$\boldsymbol{T_{off}}$	Channel contention delay due to random back-off when the channel is not idle (s)
$\boldsymbol{Q}$	Success packet transmission probability without collision occurred
$\boldsymbol{\lambda_{packet}}$	Packet arrival rate (s^{-1})

Table 6- 2: Places, Transitions, Transition Rates, Arcs and Arc Multiplicities for the SPN Model in Figure 6-1.

Place	Meaning
T_m	*mark(T_m)* means the number of trusted member nodes
UC_m	*mark(UC_m)* means the number of compromised but undetected member nodes
DC_m	*mark(DC_m)* means the number of correctly detected compromised member nodes
FDC_m	*mark(FDC_m)* means the number of falsely detected member nodes as compromised
TJ	*mark(TJ)* means the number of aggregated trusted join requests by a new member
TL	*mark(TJ)* means the number of aggregated trusted leave requests by a current member
GF	*mark(GF)* == 1 means that group security failure has occurred due to illegal data leak-out

Transition	Rate or Probability	Physical Meaning
T_CP	*A (m_c)*	A node has been compromised
T_IDS	*mark(UC_m)* D (m_d)*(1-P_{fn})*	A compromised node has been detected
T_FA	*mark(T_m)* D (m_d)*P_{fp}*	A node has been falsely diagnosed as compromised
T_RK	*1/T_{cm}*	A rekeying operation has been performed
T_DRQ1	*mark(UC_m)*p1*λ_q*	A group communication operation has been performed by compromised but not detected member nodes (*UC_m*)
T_DRQ2	*mark(DC_m)*p1*λ_q*	A group communication operation has been performed by compromised and detected member nodes (*DC_m*)
T_TJ	*(mark(T_m)+mark(UC_m)*μ*	A new node requests a join to the group
T_TL	*(mark(T_m)+mark(UC_m)*μ*	A current member node requests a leave to the group

Input arc	Multiplicity	Output arc	Multiplicity
T_m –T_CP	1	***T_CP – UC_m***	1
T_m –T_FA	1	***T_FA – DC_m***	1
UC_m –T_IDS	1	***T_IDS – DC_m***	1
DC_m –T_RK	*mark(DC_m)*	***T_DRQ – UC_m***	1
UC_m –T_DRQ	1	***T_DRQ – GF***	1
TJ – T_RK	*mark(TJ)*	***T_TJ – TJ***	1
TL – T_RK	*mark(TL)*	***T_TL – TL***	1
FDC_m – T_RK	*mark(FDC_m)*		

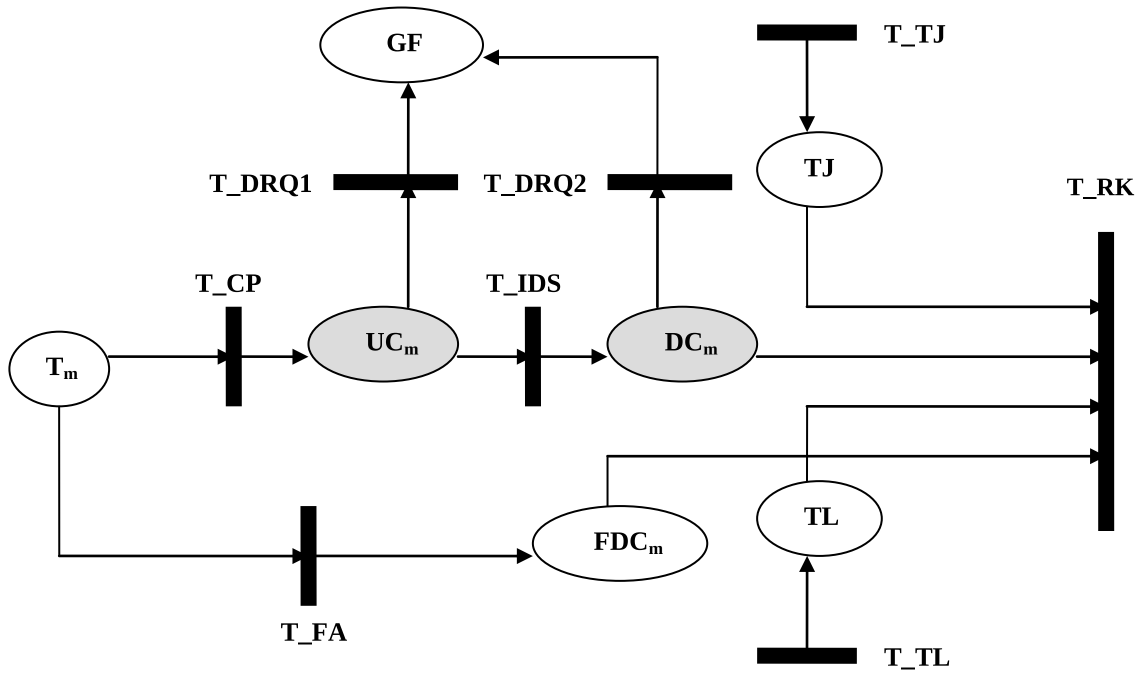


Figure 6- 1: SPN Model.

The SPN model is constructed as follows:

- We use places to classify nodes. Specifically T_m holds trusted members, UC_m holds compromised nodes that have not been detected by IDS, FDC_m holds nodes falsely diagnosed by IDS as compromised, DC_m holds compromised nodes that have been detected by IDS, *TJ* holds nodes that have issued a join request, and *TL* holds nodes that have issued a leave request.
- We use transitions to model events. All transitions in the SPN model are timed transitions. The time taken for a transition to fire depends on the event associated with it. For example, transition *T_RK* stands for a "rekeying" event so the rate at which *T_RK* fires depends on the time taken for the system to perform a rekeying operation based on GDH. As another example, transitions *T_TJ* and *T_TL* represent join and leave events, respectively, with their rates depending on the population in places T_m and UC_m, that is, $mark\ (T_m) + mark\ (UC_m)$, where *mark(X)* returns the number of tokens held in place *X*.
- We associate triggering conditions with a transition to model conditions under which an event would happen. For example, the triggering condition of *T_RK* depends on the batch rekeying technique used. For *individual rekeying*, if there is a token in FDC_m, DC_m, *TJ*, or *TL*, transition

T_RK is triggered. For *TAUDT* if either *mark(TJ)* + *mark(TL)* reaches *k1*, or *mark*(FDC_m) + *mark*(DC_m) reaches *k2*, transition *T_RK* is triggered. For *JALDT* if either *mark(TJ)* reaches *k1* or *mark(TL)*+ *mark*(FDC_m) + *mark*(DC_m) reaches *k2*, *T_RK* fires. Note that places *TJ* and *TL* are used to explicitly count the number of join and leave events to trigger transition *T_RK* according to the threshold-based periodic batch rekeying protocol selected to be executed by the system.

- We move nodes (tokens) from one place to another place when an event occurs. For example, after *T_RK* fires, all pending join/leave/eviction operations will be processed by the system. This is modeled by flushing tokens in places FDC_m, DC_m, *TJ*, and *TL*. This is achieved by specifying the "multiplicity" associated with an *arc*. For example, to evict all nodes in DC_m, the multiplicity of the arc connecting place DC_m and transition *T_RK* is *mark*(DC_m), so after *T_RK* fires all the tokens (nodes) in place DC_m are flushed, representing that *mark*(DC_m) nodes have been evicted after a rekeying operation is done. Simultaneously, all tokens (nodes) in other places FDC_m, *TJ*, and *TL* are removed as well.
- Initially, all members are trusted; thus, we place all *N* members in place T_m as tokens. Trusted members may become compromised because of insider attacks with a node-compromising rate $A\ (m_c)$. This is modeled by firing transition *T_CP* and moving one token at a time (if it exists) from place T_m to place UC_m. Tokens in place UC_m represent compromised but undetected member nodes.
- We consider the system as having experienced a security failure when data are leaked out to compromised but undetected members, i.e., due to Condition C1. Thus, when a token exists in place UC_m, the system is considered to be in a security vulnerable state. A compromised but undetected member will attempt to compromise data from other members in the group. Because of the use of host-based IDS, a node will reply to such a request only if it could not identify the requesting node as compromised with the per-node false negative probability *p1*. This is modeled by associating transition *T_DRQ1* with rate $p1*\lambda_q * mark\ (UC_m)$. The firing of transition *T_DRQ1* will move a token into place *GF*, at which point we regard the system as having experienced a security failure due to Condition C1. Specifically, when *mark(GF)* > 0, the system fails due to Condition C1, where *mark(GF)* returns the number of tokens contained in place *GF*.

- A compromised node in place UC_m may be detected by IDS before it compromises data in the GCS. The intrusion detection activity of the system is modeled by the detection function with rate $D(m_d)$. Whether the damage has been done by a compromised node before the compromised node is detected depends on the relative magnitude of the node-compromising rate ($A(m_c)$) versus the IDS detection rate ($D(m_d)$). When transition *T_IDS* fires, a token in place UC_m will be moved to place DC_m, meaning that a compromised but undetected node now becomes detected by IDS. For voting-based IDS, the transition rate of *T_IDS* is $mark\ (UC_m)*D\ (m_d)*\ (1\text{-}P_{fn})$, taking into consideration of the false negative probability of voting-based IDS used. Voting-based IDS can also false-positively identify a trusted member node as compromised. This is modeled by moving a trusted member in place T_m to place DC_m after transition *T_FA* fires with rate $mark(T_m)*D\ (m_d)*\ P_{fp}$. Note that voting-based IDS parameters, P_{fn} and P_{fp}, can be derived based on *p1* and *p2*, the number of vote-participants (*m*), and the current number of compromised nodes which may collude to disrupt the service of the system.
- After a node is detected by IDS as compromised, it is evicted when a rekeying operation is invoked, triggered either by *k1* and *k2* in a double threshold-based periodic batch rekeying protocol. This is modeled by firing transition *T_RK* for evicting detected compromised members. The rate at which transition *T_RK* fires (for performing a rekeying operation based on GDH) is $1/\ T_{cm}$. Since an evicted node (in place DC_m) does not leave the group until the next batch rekey interval period, it introduces security vulnerability because they may perform attacks using fake identities (e.g., impersonation attacks). We model this data leak-out vulnerability by a transition *T_DRQ2* connecting DC_m and *GF* with rate $p1*\lambda_q\ *\ mark\ (DC_m)$. The firing of transition *T_DRQ2* will move a token into place *GF*, at which point we regard the system as having experienced a security failure again due to Condition C1. This also models the case that while a double threshold-based periodic batch rekeying algorithm with either $k1 > 1$ or $k2 > 1$ may improve rekeying efficiency, it may expose the system to this security vulnerability.
- The GCS is characterized by member join and leave events, with rates of λ and μ, respectively. This is modeled by associating transitions *T_TJ*, and *T_TL* with these two rates.
- The system is considered as experiencing a security failure if either one of the two security failure conditions, Condition C1 or Condition C2, is met. This is modeled by making the system enter an absorbing state when either Condition C1 or Condition C2 is *true*. In the SPN

model, this is achieved by associating every transition in the SPN model with an enabling function that returns *false* (thus disabling the transition from firing) when either Condition C1 or Condition C2 is met, and *true* otherwise. In our model, Condition C1 is *true* when *mark(GF)* > 0 representing that data have been leaked out to compromised members; Condition C2 is *true* when more than 1/3 of member nodes are compromised as indicated in Equation 6-1 below, where *mark* (UC_m) returns the number of compromised but undetected nodes in the system, *mark*(DC_m) returns the number of compromised and detected nodes in the system, *mark*(FDC_m) returns the number of nodes falsely detected as compromised in the system, and *mark*(T_m) returns the number of trusted healthy nodes in the system.

$$\frac{mark(UC_m) + mark(DC_m)}{mark(T_m) + mark(UC_m) + mark(FDC_m) + mark(DC_m)} > \frac{1}{3} \qquad (6\text{-}1)$$

6.2.2 Parameterization

Here we describe the parameterization process, i.e., how to give model parameters proper values reflecting the operational and environmental conditions of the system.

- ***N*:** This is the number of current active group members in the system. This number evolves dynamically as the system evicts compromised nodes. Since a node leaves the group voluntarily with rate μ and joins the group with rate λ, the probability that a node is active in the group is $\lambda/(\lambda+\mu)$ and the probability that it is not is $\mu/(\lambda+\mu)$. Let n be the total group population at any time ($n=N_{init}$ at t=0). Then, $N = n\,\lambda/(\lambda+\mu)$. In the SPN model, we initially place $N_{init}\lambda/(\lambda+\mu)$ tokens in place T_m. As the system evolves, N is obtained with *mark* (T_m) + *mark* (UC_m) indicating the number of current active group members.
- **Λ_J & Λ_L:** These are the aggregate join and leave rates of group nodes, respectively. They are also the transition rates associated with *T_TJ* and *T_TL*. The aggregate leave rate Λ_L is equal to the number of active group members (N) multiplied by per-node join rate (μ). It is easy to see that this aggregate leave rate Λ_L by active members is the same as the aggregate join rate Λ_J by non-active group members.
- **T_{cm}:** Based on the GDH protocol, T_{cm} can be calculated as already shown in Chapter 5 (Equation 5-3).
- **$A(m_c)$:** We adopt the *linear time attacker* function ($A_{linear}(m_c)$) as shown in Chapter 5 (Equation 5-4).

- ***D (m_d)*:** We parameterize it based on *linear periodic detection* function ($D_{linear}(m_d)$) as shown in Chapter 5 (Equation 5-5).
- ***P_{fn}* & *P_{fp}*:** These two parameters follow Equation 5-6 in Chapter 5.

In this work, we use two metrics, *MTTSF* and $\bar{R}$. *MTTSF* is calculated based on Equation 5-9 in Chapter 5. The ***average service response time per group communication packet*** ($\bar{\boldsymbol{R}}$) is calculated as the sum of wireless network delay (T_b) and transmission delay (T_{com}). Specifically, $\bar{R}$ is computed as:

$$\bar{R} = \frac{\int_0^{MTTSF} [T_b(t) + T_{com}(t)]dt}{MTTSF} \qquad (6\text{-}2)$$

where

$$T_b = T_c + \left(T_c + T_{off}\right) \times (1/Q - 1)$$

$$T_c = T_{RTS} + SIFS + T_{CTS} + SIFS + DIFS$$

$$T_{off} = E[CW] \times T_{slot}$$

$$Q = e^{-\lambda_{packet} \times T_c}$$

$$T_{com} = \frac{(b_{GC} + b_{ack})}{BW}$$

Here T_{com} accounts for the transmission delay for a group communication packet being delivered to the destination, including the time to get an acknowledgement back; b_{GC} is the packet size (*bits*) of a group communication operation and b_{ack} is the packet size (*bits*) for an acknowledgement. T_b accounts for the wireless channel contention time estimated based on RTS (Request-To-Send)/CTS (Clear-To-Send) protocols in IEEE 802.11 with DCF (Distributed Coordination Function). The contention time depends on the number of retries for securing the wireless channel. Each trial has a basic delay of T_c including the transmission time of the RTS and CTS packets plus the artificial delay (SIFS and DIFS) intrinsic to IEEE 802.11. If a trial is not successful, there is a backoff time T_{off} before the next trial is taken place. While in practice the backoff window size is randomly determined over a range, to simplify our analysis we assume the average window size, denoted by *E [CW]*, is being used in each trial. The values used for T_{RTS}, T_{CTS}, *SIFS*, *DIFS*, and T_{slot} as shown in Table 6-3 are based on DSSS (Direct Sequence

Spread Spectrum) for IEEE 802.11 as reported in [10, 16]. An attempt is successful if there is no other packet being transmitted during the RTS/CTS sequence. Since the overall packet rate is λ_{packet}, assuming packets arrive in accordance with a Poisson process, the probability of no packet arrival during T_c, or the probability of no collision, is given by $e^{-\lambda_{packet} \times T_c}$. By modeling the channel contention process as a geometric distribution with success probability Q, the average number of tries before a successful transmission without collision is obtained by *1/Q*. Note that here we ignore the very small propagation delay (i.e., 2 *micros*) in calculating T_b.

6.3 Numerical Results

We present numerical results obtained from evaluating the SPN model developed and provide physical interpretations. Our objective is to identify optimal settings in terms of optimal double thresholds *k1* and *k2* of batch rekeying protocols and optimal intrusion detection intervals that maximize *MTTSF* while satisfying performance requirements in terms of service response time ($\bar{R}$). In particular, based on the identified optimal *k1* and *k2* thresholds, optimal intrusion detection intervals are identified. We compare the system performance of double threshold-based periodic batch rekeying protocols against the baseline *individual rekeying* integrated with voting-based IDS.

Table 6- 3: Parameters and Default Values.

Parameter	Value	Parameter	Value
λ	1/(60*60 s)	***m***	5
μ	1/(60*60*4 s)	***BW***	1*Mbps*
T_{IDS}	30 – 9600 *s*	N_{init}	60 nodes
T_{status}	2 *s*	***D*** (m_d)	Linear to m_d
λ_c	1/(60*60*12 s)	***A*** (m_c)	Linear to m_c
λ_q	1/(60*30 s)	T_{RTS}	0.0003 *s*
p1	1 %	T_{CTS}	0.0004 *s*
p2	1 %	***SIFS***	0.00002 *s*
b_{GDH}	64 *bits*	***DIFS***	0.00005 *s*
b_{GC}	800 *bits*	T_{slot}	0.00005 *s*
b_{ack}	32 *bits*	***E[CW]***	256

We vary certain design parameters to analyze their effects on system performance. Table 6-3 summarizes default parameter values. In particular, we use *p1* = *p2* = 1% since in general less than 1% of false positive or false negative rate is deemed acceptance. For voting-based IDS, P_{fn} and P_{fp} are calculated based on Equation 5-6.

6.3.1 Optimal Double Thresholds (*k1* and *k2*)

Figures 6-2 and 6-3 show the effect of varying *k1* and *k2* on *MTTSF* for *TAUDT* and *JALDT*, respectively. The optimal *MTTSF* in *TAUDT* is observed at (*k1*, *k2*) = (4, 1), as shown in Figure 6-2. We explain why the optimal (*k1*, *k2*) = (4, 1) under *TAUDT* below. Recall that in *TAUDT*, *k1* governs against the number of join/leave nodes (*mark(TJ)* + *mark(TL)*) while *k2* governs against the number of nodes detected as compromised (*mark(FDC$_m$)* + *mark(DC$_m$)*). As *k2* increases, security failure due to Condition C1 is more likely to occur since a larger *k2* allows more detected compromised nodes to exist. Allowing *k2* larger than 1 significantly deteriorates *MTTSF*. Thus, *k2* is optimized at 1. When *k1*=1, the probability that rekeying is triggered due to *k1* is relatively high compared to when *k1* > 1. This has the effect of delaying detected compromised nodes (in *DC$_m$*) to be removed, which degrades *MTTSF* again due to Condition C1. As *k1* increases, the probability that rekeying is triggered due to *k2* increases. This has the effect of quickly removing detected compromised nodes, which increases *MTTSF* as a result. Lastly, as *k1* increases further, not only nodes in *DC$_m$* but also nodes in *FDC$_m$* are very quickly removed. This has the effect of degrading *MTTSF* due to Condition C2. We also note that when *k2* is greater than 1, there is not much sensitivity of *MTTSF* on *k2* since *k2* governs untrusted members directly related to security failure.

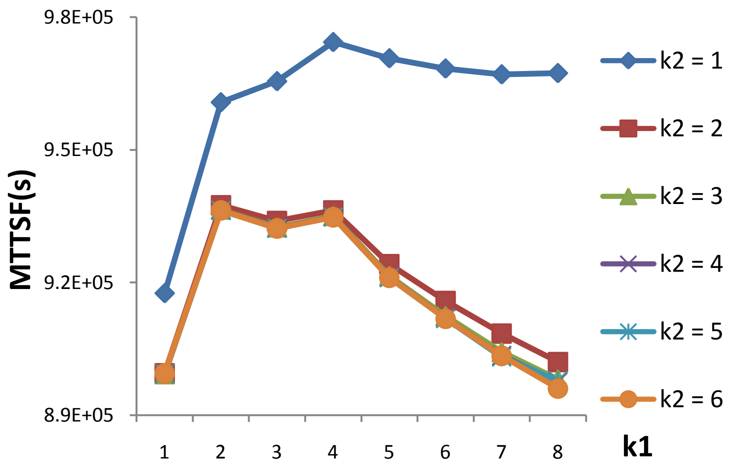


Figure 6- 2: Optimal *k1* and *k2* for *TAUDT* in *MTTSF*.

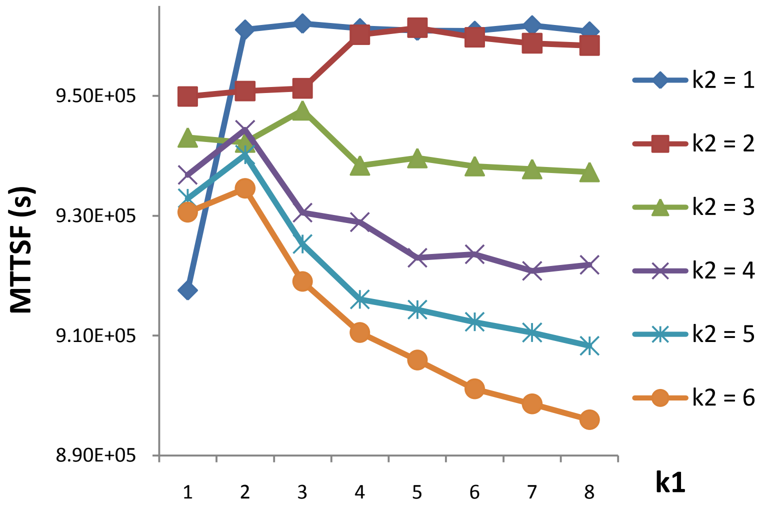


Figure 6- 3: Optimal *k1* and *k2* for *JALDT* in *MTTSF*.

The optimal *MTTSF* in *JALDT* is observed at (*k1*, *k2*) = (5, 2), as shown in Figure 6-3. Recall that in *JALDT k2* governs the threshold for both trusted leave and untrusted leave requests, while in *TAUDT k2* only governs untrusted leave requests. Consequently, the optimal *k2* is at 2 in *JALDT* as opposed to the optimal *k2* at 1 in *TAUDT*. The reason of having the optimal *k1* = 5 in *JALDT* is that *k1*=5 (as opposed to 4) best balances the probability of security failure due to Condition C1 versus Condition C2, as explained earlier, since *k1* now only governs join operations.

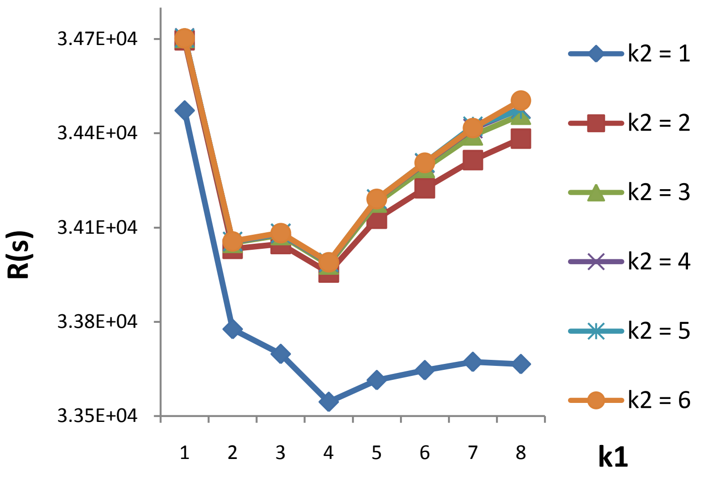


Figure 6- 4: Optimal *k1* and *k2* for *TAUDT* in Service Response Time $\bar{R}$.

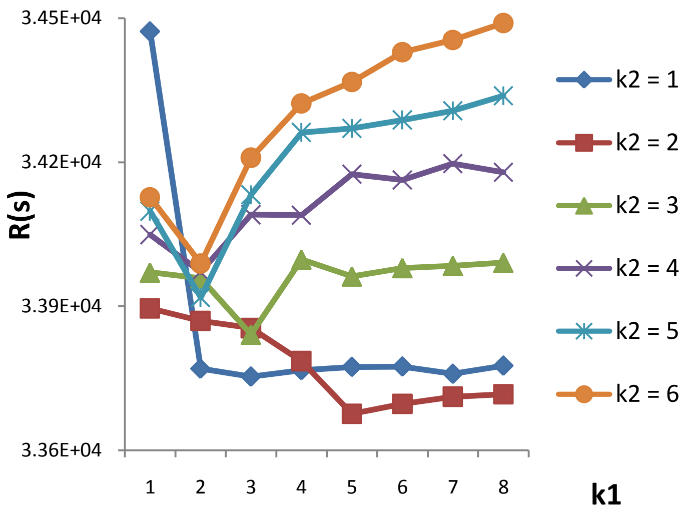


Figure 6- 5: Optimal *k1* and *k2* for *JALDT* in Service Response Time $\bar{R}$.

Figures 6-4 and 6-5 show the effect of *k1* and *k2* on the average service response time, $\bar{R}$. The trends shown in Figures 6-4 and 6-5 strikingly reflect the overall communication cost per time unit (*s*) versus *k1* and *k2* (not shown here for brevity). In Figure 6-4, we see the optimal (*k1, k2*) is at (4, 1) being identical to that in Figure 6-2. In Figure 6-5, we also observe that the optimal (*k1, k2*) is at (5, 2) being identical to that in Figure 6-3. The reason of having the optimal point at (4, 1) in *TAUDT* is due to the tradeoff between rekeying traffic versus group communication traffic. First, when *k1* is low, the traffic introduced due to frequent rekeying is high, which results in high $\bar{R}$. As *k1* increases, the traffic introduced due to rekeying decreases, so $\bar{R}$ also decreases. However, as *k1* increases further, $\bar{R}$ increases again because of increased traffic introduced by group communication since more members are allowed to stay in the system under a high *k1* value. Consequently *k1* = 4 becomes the optimal *k1* value. On the other hand, since *k2* governs how many compromised nodes can be kept in the system until rekeying, higher k2 implies a higher traffic introduced by group communication. Such a traffic cost outweighs the traffic cost introduced due to rekeying. Consequently, $\bar{R}$ increases as *k2* increases, resulting in the lowest $\bar{R}$ at *k2* = 1. The reason of having the optimal point at (5, 2) in *JALDT* can be explained in a similar way.

6.3.2 Optimal Intrusion Detection Intervals (T_{IDS})

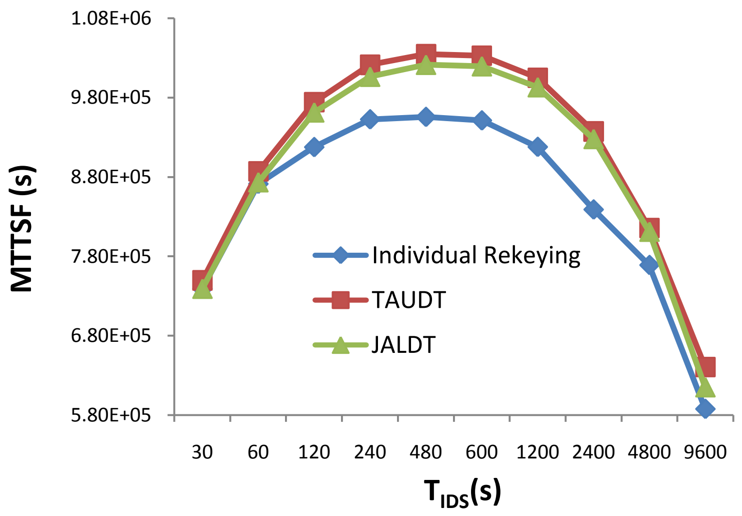


Figure 6- 6: Optimal T_{IDS} in *MTTSF*.

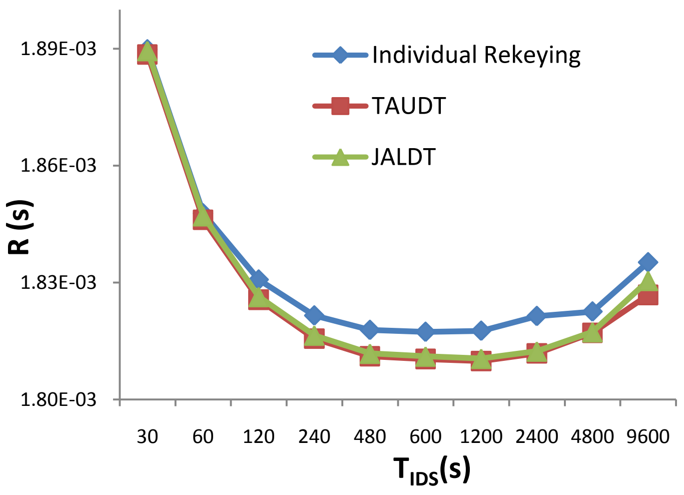


Figure 6- 7: Optimal T_{IDS} in $\overline{R}$.

Next we analyze optimal intrusion detection intervals (T_{IDS}) based on optimal double thresholds *k1* and *k2* identified, that is, for *TAUDT*, (*k1*, *k2*) = (4, 1) and for *JALDT*, (*k1*, *k2*) = (5, 2) for all T_{IDS} ranges respectively. We compare system performance under periodic batch rekeying versus individual rekeying and show that batch rekeying under optimal settings outperforms individual rekeying when IDS is present.

Figure 6-6 shows the effect of three different rekeying protocols on *MTTSF* and identifies the optimal intrusion detection interval, T_{IDS}. We observe that there exists an optimal T_{IDS} that maximizes *MTTSF*. In general, as T_{IDS} increases, *MTTSF* increases until its optimal T_{IDS} is reached, and then *MTTSF* decreases after the optimal T_{IDS}. The reason of decreasing *MTTSF* after reaching the optimal point is that the false positive probability (P_{fp}) increases as T_{IDS} decreases, resulting in more nodes being falsely identified as compromised and being evicted from the system. Note that P_{fp} is one aspect of false alarms generated by IDS, so its effect is more increased when IDS is more frequently triggered. As expected, we observe that the baseline individual rekeying performs the worst, while *TAUDT* performs the best in terms of *MTTSF* among the three. Here *TAUDT* operates at the optimal setting (*k1*, *k2*) = (4, 1) as identified earlier. On the one hand, *k2* =1 allows rekeying to be triggered as soon as possible once a compromised node has been identified for eviction. On the other hand, *k1*=4 balances the probability of security failure due to Condition C1 versus Condition C2, as explained earlier. We note that i*ndividual rekeying* performs the worst because the probability that rekeying is triggered due to trusted join/leave is relatively high compared to the other two rekeying protocols. This has the effect of removing detected compromised nodes in DC_m slowly and decreasing *MTTSF* due to Condition C1. The optimal intrusion detection interval is identified at T_{IDS} = 240 *s* for individual rekeying, and 480 *s* for *TAUDT* and *JALDT*, as shown in Figure 6-6.

Figure 6-7 shows service response time ($\bar{R}$) versus intrusion detection interval (T_{IDS}). We again observe that there exists an optimal T_{IDS} that minimizes the service response time in all three curves. The reason of having an optimal point that minimizes $\bar{R}$ is attributed to the tradeoff between the IDS traffic versus group communication traffic. That is, when T_{IDS} is sufficiently low (T_{IDS} < 600 *s*), the IDS traffic is high due to the high frequency of triggering IDS, thus resulting in high $\bar{R}$. As T_{IDS} increases, on the other hand, the IDS traffic decreases but the group communication traffic substantially increases since more compromised members not detected will stay in the system and participate in group communication activities. Consequently, this tradeoff results in the optimal T_{IDS} being at 600 *s*. Among three curves in Figure 6-7, we again observe that *individual rekeying* performs the worst due to the frequent triggering of rekeying operations upon every membership change, while *TAUDT* at the optimal point performs the best, but showing almost same performance in *JALDT*.

A system designer can use the results obtained here to identify T_{IDS} that can optimize system performance. To maximize *MTTSF*, T_{IDS} is identified at 480 *s*. To minimize $\bar{R}$, T_{IDS} is identified at 600 *s*. However, there is an insignificant response time difference between T_{IDS} = 480 *s* and T_{IDS} = 600 *s*. Thus, the optimal T_{IDS} in this case is set to 480 *s* that can maximize *MTTSF* while satisfying the average service response time ($\bar{R}$) requirement.

6.4 Summary

In this chapter, we investigated the design of integrating intrusion detection with batch rekeying to cope with both outsider and insider attacks for GCSs in MANETs, and analyzed the tradeoff between security and performance properties of the resulting GCS due to the use of these two protocols. We showed that there exist optimal settings in terms of rekeying thresholds (*k1* and *k2*) reflecting batch rekey intervals and intrusion detection intervals under which the system lifetime in terms of *MTTSF* can be maximized while satisfying imposed performance requirements in terms of average service response time, $\bar{R}$. We also demonstrated the effectiveness of the approach by comparing the resulting system performance with that of a system with IDS integrated with individual rekeying.

Chapter 7 REGION-BASED GROUP KEY MANAGEMENT

In this chapter, we propose and analyze region-based group key management protocols for scalable and reconfigurable group key management in MANETs. The novelty of these protocols lies in that they are QoS-aware and designed to optimize performance while satisfying security requirements in secrecy, availability and survivability. The content is largely based on our published conference paper [18] and journal paper [22].

In Section 7.1, the background and protocol description are discussed. In Section 7.2 we develop a performance model to evaluate the proposed region-based group key management protocol and explain how we parameterize model parameters for characterizing the operational conditions of mobile group communications in MANETs, considering all possible events induced by group management including group leave/join/eviction and group merge/partition. Section 7.3 presents numerical results obtained from evaluating the performance model, and gives the physical interpretation of the results obtained. A simulation study was conducted to validate analytical results reported. Section 7.4 concludes this chapter.

7.1 Region-based Group Key Management Protocol

7.1.1 Background

Conceivably, as the number of group members becomes large, group key management can incur significant communication overheads and cause significant performance degradation. We propose a reliable and secure region-based group key management protocol for secure group communications in MANETs. For scalability and dynamic management, we propose a two-level hierarchical key management protocol following the IETF Group Key Management Architecture [114] to efficiently and securely distribute keys and the *CKA* protocol [1, 2, 3, 54] for key generation. The protocol is designed for MANETs with no infrastructure support. It is QoS-aware in that it embeds designs to allow optimal settings to be identified and applied at runtime to minimize the overall communication cost due to group key management while satisfying the secrecy/security requirements of the system, when given a set of parameter values characterizing the operational and environmental conditions of a GCS in MANETs.

For scalability and efficiency, our region-based group key management protocol divides a group into region-based subgroups based on *decentralized* key management principles as illustrated in Figure 7-1. We assume that each group member is equipped with GPS and knows its location as it moves across regions. When a regional boundary is crossed, a member retains its group membership, but changes its subgroup "regional" membership. For secure group communications, all group members share a secret group key, K_G. On the other hand, for secure subgroup communications, all subgroup members in region *i* share a secret key, K_{Ri}. For robustness, a *distributed* key management protocol is used to generate and manage the shared secret key. In the discussion below, we assume that a *CKA* protocol such as *GDH* is used for this purpose. The regional area size is an important parameter that determines the cost of group key management. Our proposed region-based group key management protocol will operate at the optimal regional area size identified to minimize the cost of key management in terms of network traffic.

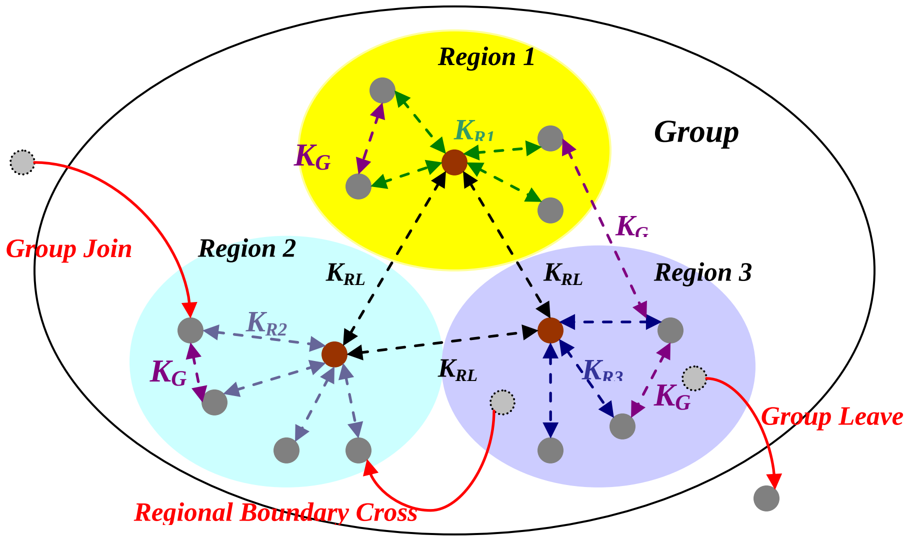


Figure 7-1: Region-Based Group Key Management.

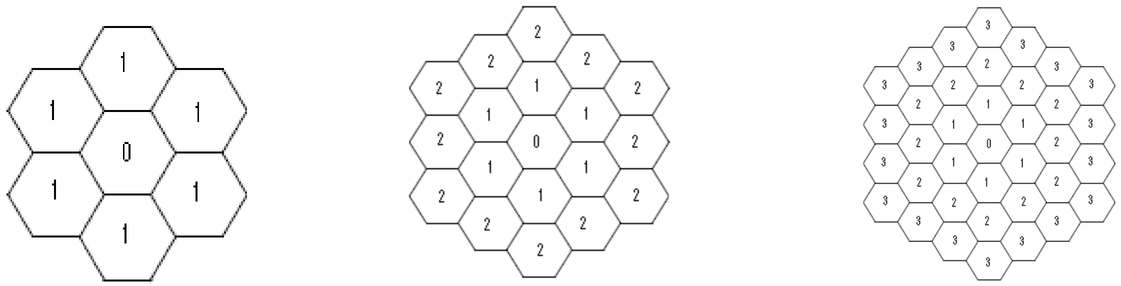
Assume that nodes are randomly distributed according to a homogeneous spatial *Poisson* process with node density λ_p. Assume that the operational area is $A = \pi r^2$, where r is the radius of the operational area. Thus, the average number of nodes in the system is $N=\lambda_p A$. Recall our assumption on join/leave events that a node may leave a group voluntarily with rate μ any group with rate λ due to tactical reasons, as we assumed in Chapters 4 and 5. Then, the probability that a node is in any group is $\lambda/(\lambda+\mu)$ and the probability that it is not in any group is $\mu/(\lambda+\mu)$. It follows that the average number of nodes that are active members is given by *N*. Furthermore, let Λ_J and Λ_L be the *aggregate* join and leave rates of all nodes, respectively. Then, Λ_J and Λ_L, can be calculated as (already showed the same equation in Equation 5-2 to calculate the aggregated group join and leave rate in equilibrium):

$$\Lambda_J = \lambda \times N \times \frac{\mu}{(\lambda + \mu)}, \qquad \Lambda_L = \mu \times N \times \frac{\lambda}{(\lambda + \mu)} \qquad (7\text{-}1)$$

Nodes can move freely with a mobility rate of σ. Nodes that are connected with each other form a group. When all nodes are connected, there is only a single group in the system. Due to node mobility, a group may be partitioned into two. Conversely, two groups may merge into one as connectivity resumes. We assume that the secure GCS is designed to support a mission critical application. All nodes are charged to complete a *mission* and the mission critical application allows group merging and partitioning activities in response to network dynamics. However, a group, no matter of its size, acts independently of other groups to complete the mission. Nodes in a group must satisfy the *forward/backward secrecy*, *confidentiality*, *integrity* and *authentication* requirements for secure group communications in the presence of malicious outside attackers.

Reliable transmission is a system requirement for secure group communications. This can be achieved by using ACK packets and packet retransmission upon timeout. The centralized approach has a single point of failure and is not acceptable for large GCSs. This work aims to design and analyze a distributed group key management protocol that is scalable to large systems and is robust to node/network failure and group partition, while minimizing the network communication cost.

We use a hexagon to model a region [106]. Figure 7-2 shows a case in which the operational geographical area $A = \pi r^2$ is divided into $3n^2 + 3n + 1$ = 37 regions with n = 3, 19 regions with n = 2, and 7 regions with n = 1. Let $R(n)$ denote the number of regions (i.e., $3n^2 + 3n + 1$) in the operational area. The expression for $R(n) = 3n^2 + 3n + 1$ is derived by mathematical induction based on the hexagonal network coverage model used.



***n* = 1**
7 regions

***n* = 2**
19 regions

***n* = 3**
37 regions

Figure 7-2: A Geographic Area Divided into $3n^2 + 3n + 1$ Hexagons with *n* =1, 2, and 3.

A member can move around by crossing boundaries between regions. Assume that members are always confined in the geographical area of πr^2 divided into $R(n) = 3n^2 + 3n + 1$ regions as in a battlefield situation. The total number of regional boundary edges is $6(3n^2 + 3n + 1)$ counting internal edges twice to include reverse traffic. The total number of outward boundary edges surrounding the geographical area is $12n + 6$. The probability that a member moves across a boundary between two regions (but not going out of the geographical area) once a move is made, $P_{RM}(n)$, is given by:

$$P_{RM}(n) = \frac{6(3n^2 + 3n + 1) - (12n + 6)}{6(3n^2 + 3n + 1)} \qquad (7\text{-}2)$$

The mobility model of a node remains the same. However, the mobility rate of a node, defined as the rate at which a region is crossed, changes depending on the number of regions $R(n)$ in the operational area. Let the mobility rate of a node be σ when there is only one region. As we divide the area into more regions (i.e., from $R(n)$ = 1, 7 to 19, and so on as we increase n from 0, 1 to 2), the "regional" mobility rate increases because as the regional size decreases causing more boundary-crossing events to occur per time unit. Let σ_n be the *regional mobility rate* when there are $R(n)$ regions. Then $\sigma_n = (2n+1)\, \sigma\, P_{RM}(n)$ because a node would cross $2n$+1 regions when there are $R(n)$ regions for the same amount of time it would take to cross a regional boundary when there is only one region in the geographical area. The factor $P_{RM}(n)$ is multiplied to account for the fact that not all moves will cross a regional boundary. The relationship between n, $R(n)$ and σ_n is summarized below:

$$\begin{array}{lll} level\ 0 & 1\ hexagon & \sigma_0 = \sigma \\ level\ 1 & 7\ hexagons & \sigma_1 = 3\sigma P_{RM}(1) \\ level\ 2 & 19\ hexagons & \sigma_2 = 5\sigma P_{RM}(2) \\ & \vdots & \\ level\ n & (3n^2+3n+1) hexagons & \sigma_n = (2n+1)\sigma P_{RM}(n) \end{array} \tag{7-3}$$

7.1.2 Protocol Description

Below we describe how our region-based group key management protocol works in MANETs.

Bootstrapping: In the initial bootstrapping process, a node within a region can take the role of a regional "leader" to perform GDH. If there are multiple initiators, then the node with the smallest *id* will win as the leader and will execute GDH to completion to generate a regional key. A key agreement protocol such as GDH is robust such that if a node leaves or moves out of the region during the execution of GDH, the remaining nodes can re-execute the protocol to completion and eventually agree on a shared secret key. Once a leader is generated in each region, all leaders in the group will execute GDH to agree on a secret *leader key*, K_{RL}, for secure communications among leaders. Similar with the bootstrapping process within a region, a "leader" can take the role of a "super-leader" or "coordinator" to execute GDH among leaders. If there are multiple leaders initiating the execution of GDH, the leader with the smallest *id* will win as the coordinator to execute GDH to completion to generate K_{RL}. Once K_{RL} is generated, a *group key*, K_G, is derived by means of K_G = MAC (K_{RL}, c), where MAC is a cryptographically secure hash function, K_{RL} is the leader key used as the secret key to MAC, and c is a fresh counter which will be incremented whenever a group membership event occurs. Once K_G is generated, leaders will disseminate the group key K_G to group members in their regions. The group key K_G then is used for secure data communications among group members across regions. We note that a leader is responsible for first generating a new group key based on the current leader key (agreed upon by all leaders) and then broadcasting the renewed group key to members in its region. All leaders do the same, so the renewed group key may be disseminated to all members in all regions. In this sense, the leader is carrying out a centralized rekeying protocol in its region since the generation of the renewed key is done by the leader alone based on the current leader key without having to coordinate with other leaders.

Key Management: These shared secret keys at the subgroup (regional), leader, group levels may be rekeyed to preserve secrecy in response to events that occur in the system. The *leader key* (K_{RL}) is rekeyed whenever there is a leader change, including a leader crossing a regional boundary or leaving the group, a leader failure, and a group merge or partition event. The *regional key* (K_R) is rekeyed whenever there is a regional membership change, including a local member group join/leave, a node failure, a local regional boundary crossing, and a group merge or partition event.

Table 7- 1: Notation.

Symbol	Meaning
K_G	Group key
K_{RL}	Leader key
K_{Ri}	Regional key in region *i*
RV_i	Regional view in region *i*
LV	Leader view
GV	Group view
RL_i	A leader in region *i*
$RM_{j,i}$	A member *j* in region *i*

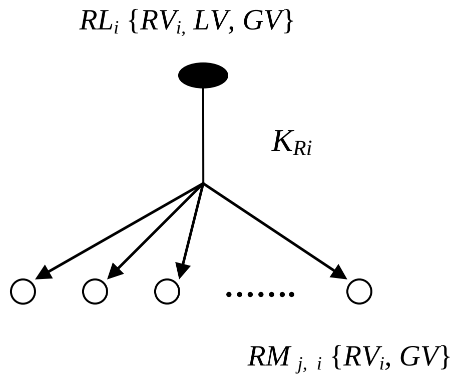


Figure 7-3: Views for Leaders and Members.

View Management: In addition to maintaining secrecy, our region-based group key management protocol also allows membership consistency [106] to be maintained through *membership views*. Three membership views can be maintained by various parties: (a) *Regional View (RV)* contains regional membership information including regional (or subgroup) members' *id*s and their location information, (b) *Leader View (LV)* contains leaders' *id*s and their location information, and (c) *Group View (GV)* contains group membership information that includes members' *id*s and their location information.

Figure 7-3 illustrates the views maintained by a leader versus those by a group member. A view can be established after a shared secret key is established at the corresponding regional, leader or group level.

Rekeying Protocol: In addition to group member join/leave events which cause rekeying of the group key, mobility-induced events may also cause rekeying. Below we describe our region-based group key management protocol for a MANET in response to events that may occur in the system.

- *Group member join*: When a new member, say *A*, joins the group, *A* beacons a "*hello*" message including its *id* and location information to inform its intention to join the group. Neighboring nodes receiving the beacon forward the "*hello*" message to their regional leader. The regional leader authenticates *A*'s identity based on *A*'s public key. Then, the leader acts as a coordinator involving all subgroup members including *A* to execute GDH to generate a new regional key. The leader then updates the regional membership list, and broadcasts the regional membership list to members in the region. Since a join event incurs a group membership change, the group key is also rekeyed. The regional leader informs the newly joined member's information to all other leaders so that all leaders apply K_G = MAC (K_{RL}, c) to generate a new group key, using the current leader key K_{RL} as the secret key to MAC. All leaders then simultaneously distribute the new group key to members in their regions by encrypting the group key with their respective regional key K_R. In summary, when a new member joins the group, a regional key and a group key are rekeyed. The regional view of the region in which the group join event is initiated and the group view is updated.

- *Group member leave*: When a non-leader member, say *B*, leaves the group, it informs its leaving intention to its regional leader. When the leader receives the leaving intention message from *B*, it updates its regional view and disseminates the updated regional view to its members. Since a group leave event instigates a regional membership change event, a new regional key is generated by executing GDH and distributed to the regional members. Next, the leader informs the membership change information to all other leaders. After all leaders receive the information on the current leave event, they also multicast the changed group view to all their members.

Finally, all leaders autonomously regenerate a group key and distribute it to their corresponding members by encrypting the group key with their respective regional key K_R.

- *Group member leave by a leader member*: When a leader (who is also a member) leaves the group, a leader key also should be changed. Thus, in addition to all operations required in the above case for the non-leader member leave, a new leader is elected to replace the leaving leader. Since this involves a leader membership change, all leaders including the newly elected leader will execute GDH to generate a new leader key. Then each leader autonomously generates a new group key and distributes it to its members using the respective regional key K_R.

- *Boundary crossing by a non-leader member*: If a non-leader member crosses a regional boundary, for example, from region *i* to region *j*, a regional membership change occurs in both regions *i* and *j*. Thus, the regional keys in the two involved regions are respectively rekeyed based on GDH and the members' regional views in these two regions are updated. Since the mobility event changes neither the leader view nor the group view, no leader or group view updates are necessary. No rekeying of the group key is needed because the member leaving a region (subgroup) is still a member of the group.

- *Boundary crossing by a leader member*: If a leader member crosses a regional boundary from *i* to *j*, there is a leadership change in addition to all operations considered in the event of boundary crossing by a non-leader member. Thus, as in the group member leave by a leader member event, a new leader in the departing region is elected, the leader key is rekeyed among all leaders, and the leader view is updated among all leaders.

- *Group member disconnection and reconnection:* Members may be disconnected voluntarily (i.e., turn power off for energy saving) or involuntarily (i.e., obstructions or jamming). To detect a member failure in the group, each mobile host periodically sends an "I-am-alive" beaconing message to its leader so that the leader is aware of which members are in its region. If a leader does not hear the beacon for a certain period of time (*T*) from a member, it considers the member being disconnected. Such disconnections are treated as group leave events in our protocol. If the member being disconnected is a leader, a new leader is elected by following a new leader election protocol. Temporarily disconnected member nodes can be later reconnected and rejoin the group. Reconnections are treated as group join events.

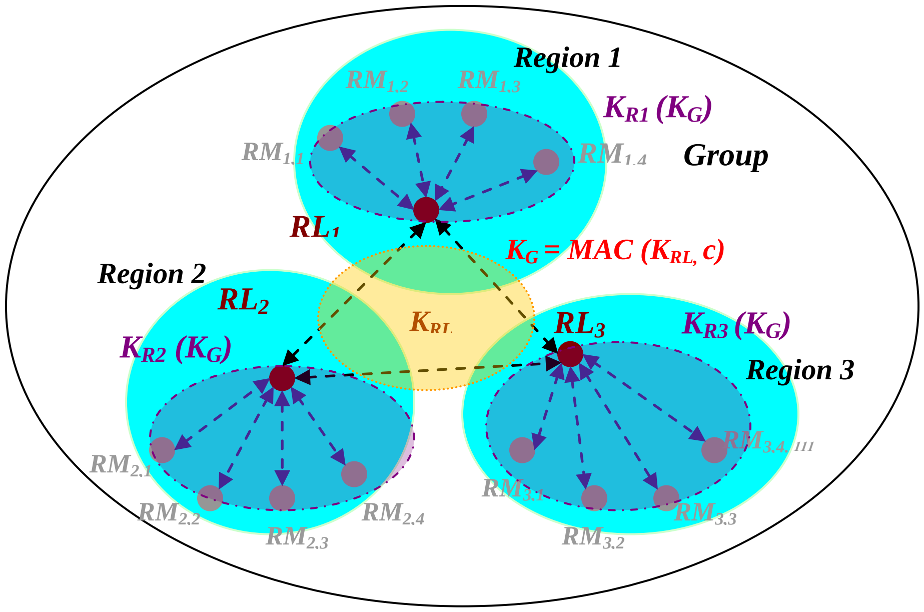
- *Leader election*: A group leave, a boundary crossing, or a disconnection by a leader member triggers a new leader election in the involved region. A member in the region after missing its regional leader's beaconing message can initiate the execution of GDH based on its regional view. If there are more than one leader invoking GDH, the member with the smallest *id* wins and will execute GDH to completion to generate a new regional key K_R. The new leader then announces itself as a new leader in the region by broadcasting a beaconing message "I-am-a-new-leader" along with the new regional view encrypted with the regional key K_R.

- *Group partition*: Group members may not be able to communicate with each other because of group partition due to node failure and node mobility. Thus, a group may be partitioned into multiple groups dynamically. Group partitioning rate increases as node density decreases and as node mobility increases. A group partition event starts with a region being partitioned and members in a region miss beaconing messages of each other. It is detected by members in a region missing the leader's beaconing message, and by the leader's missing its regional members' beaconing messages. In the former case, a new leader is elected following the "leader election" protocol discussed earlier. In the latter case, the leader with the remaining nodes in the partitioned region will execute GDH to agree on a new regional key K_R as if the nodes in the other partitioned region had been disconnected. In either case, in each partitioned group, all leaders will execute GDH to agree on a new leader key K_{RL} following the *"group member leave by a leader member"* protocol discussed earlier, as if all leaders in the partitioned group had left the group.

- *Group merge*: Two groups may merge into one when connectivity resumes. A group merge event is detected by members within a region detecting the presence of beaconing messages by non-group members. After authentication (through the two leaders), members in the merged region will execute GDH to agree on a new regional key K_R following the "*group member join*" protocol as if members in the merged region had just newly joined the group. The new leader in the merged region then coordinates with all other leaders to execute GDH to agree on a new leader key K_{RL} as if leaders had been reconnected. Finally, a new group key is generated by all leaders and is distributed to all group members in the merged group.

Group Communication Protocol: For normal group communication, we adopt to use the publish/subscribe service. We assume that all members are interested in all published data by all members. Thus, all published data in each member are disseminated to all members whenever each node publishes its data. By taking two-level hierarchical key management structure, the published data in each node is multicast to its members in the region, and then the leader receiving the published data distributes it to other leaders. After then, each leader broadcasts the published data to its members respectively. When all published data are disseminated to all members in this way, a group key is used to encrypt/decrypt the published data.

Figure 7-4 illustrates how our protocol generates and distributes regional, leader and group keys at the regional, leader and group levels, respectively. The scenario shows that there are three regions in a group, each with a regional key $K_{Ri,}$ generated through the execution of a *CKA* by the leader RL_i with its members $RM_{i,j}$ in the region. Subsequently, RL_1, RL_2, and RL_3 are engaged in *CKA* to generate a leader key K_{RL}. The group key is then generated by MAC (K_{RL}, c).



🟨 ***RL_1, RL_2, and RL_3 engaged in CKA to generate a leader key K_{RL}.***

🟪 ***RL_i and its members, $RM_{i,j}$, engaged in CKA to generate a regional key K_{Ri} .***

Figure 7-4: Key Generation and Distribution.

A group is connection-oriented such that group members must maintain connectivity for them to be in the same group. The GCS, on the other hand, is mission-oriented. Due to network partitioning caused by node mobility, a group may be partitioned into several groups. However, partitioned groups will each continue with the same mission execution. Later when two or more partitioned groups are merged into one, the merged group will still continue with the same mission execution. Therefore, mission execution is an application-level goal built on top of connection-oriented group communications. At any time, a node may belong to only one group despite group partition/merge. This definition is different from prior work that considers the possibility of a node belonging to several groups, e.g., as in [47] that attempts to develop techniques such as packing the key trees of different groups into key bundles or key parcels for performance optimization.

7.2 Performance Analysis

7.2.1 Performance Model

We develop a performance model to evaluate the network traffic cost generated for group key management in the proposed region-based protocol for MANETs. The performance analysis helps identify the optimal regional size that will minimize the network traffic generated while satisfying security properties in terms of secrecy, availability and survivability. The basic idea is to utilize the performance model developed to derive a formula to calculate the generated network traffic as a function of the regional area size, from which we could decide the best regional area size to minimize the network traffic, when given a set of basic parameter values characterizing the network and operational conditions.

The cost metric used for measuring the proposed group key management protocol is the *total network traffic per time unit* ($\hat{C}_{total}$) incurred in response to group key management events including regional mobility induced, group join/leave, periodic beacon, and group merge/partition events. The "*cost*" refers to the amount of *information bits* multiplied by the number of hops these information bits travel, i.e., hop-bits. Thus, the total cost ($\hat{C}_{total}$) consists of four components:

- ***Group Merge/Partition Cost* ($\hat{C}_{mp}$)**: this is the cost per time unit for dealing with group partition and merge events. Whenever a group partition/merge event occurs, it is required to rekey the group key and update the view for involved partitioned/merged groups such that the secrecy requirements are satisfied.
- ***Regional Mobility Cost* ($\hat{C}_{mobility}$)**: this is the cost per time unit in response to mobility-induced regional boundary crossing events, or *regional mobility handoff* events.
- ***Group Join/Leave Cost* ($\hat{C}_{join/leave}$)**: this is the cost per time unit for handling group join or leave events. This cost also includes the cost caused by connection/disconnection events by group members.

***- Periodic Beaconing Cost* ($\hat{C}_{beacon}$)**: this is the cost per time unit for maintaining view consistency by all members through periodic beaconing. Thus, this cost includes the cost for broadcasting periodic beaconing messages such as "I-am-alive" and "I-am-a-new-leader." By using this protocol, member connection or disconnection events can be detected.

***- Group Communication Cost* ($\hat{C}_{GC}$)**: this is the cost per time unit for communicating between group members. By using the publish/subscribe service, all members are subscribed to published data by all members and published data need to be propagated to all members. Thus, this cost covers respective broadcasting cost at both leader level and regional level.

The magnitude of these cost components actually depends on how many groups exist when an event occurs. To this end, we first decide the steady state probability of the system having *i* groups. Then, we calculate the *average* cost based on the steady-state probability. We use a *birth-death* process shown in Figure 7-5, to model the system. We assume that group merge and partition events follow the one-event assumption, that is, they occur one at a time.

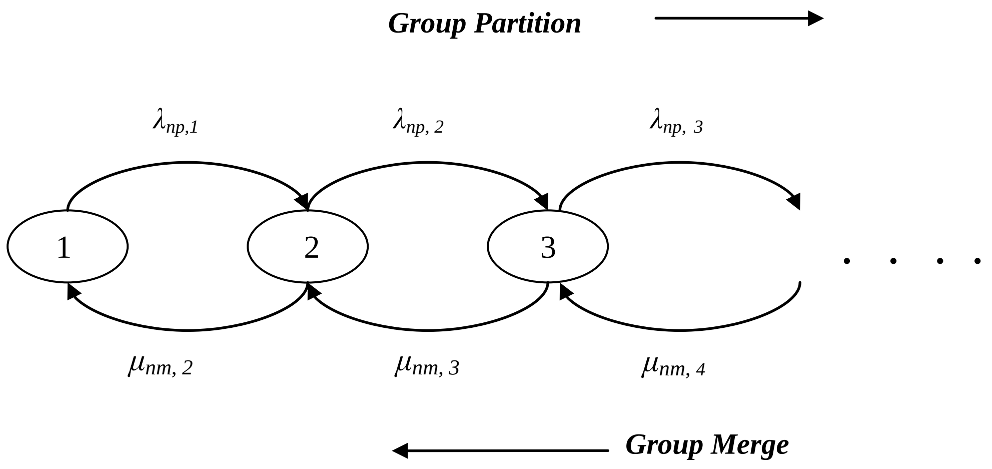


Figure 7-5: A Birth-Death Process for Modeling Group Merge/Partition.

In Figure 7-5, each state *i* represents *i* partitioned groups, at which the merging and partitioning rates are state dependent and are represented by $\mu_{nm,\,i}$ and $\lambda_{np\ i}$, respectively. These state-dependent merging/partitioning rates essentially depend on node density (λ_p), number of groups (that is, *i*), and node mobility (σ). We will explain how we parameterize these state-dependent merging/partitioning rates in Section 6.2.7. Once we parameterize the birth-death model, the probability of the system being in state *i*, Pr_i, can be easily calculated from queuing theory.

The total cost ($\hat{C}_{total}$) incurred by the region-based group key management protocol is calculated as follows:

$$\hat{C}_{total} = \hat{C}_{join/leave} + \hat{C}_{mobility} + \hat{C}_{beacon} + \hat{C}_{mp} + \hat{C}_{GC} \qquad (7\text{-}4)$$

where:

$$\begin{aligned} \hat{C}_{join/leave} &= \sum_{i=1}^{\infty} Pr_i \times \hat{C}_{join/leave,i} \\ \hat{C}_{mobility} &= \sum_{i=1}^{\infty} Pr_i \times \hat{C}_{mobility,i} \\ \hat{C}_{beacon} &= \sum_{i=1}^{\infty} Pr_i \times \hat{C}_{beacon,i} \\ \hat{C}_{mp} &= \sum_{i=1}^{\infty} Pr_i \times \hat{C}_{mp,i} \\ \hat{C}_{GC} &= \sum_{i=1}^{\infty} Pr_i \times \hat{C}_{GC,i} \end{aligned} \qquad (7\text{-}5)$$

where Pr_i is the steady-state probability that the system is in state i (i.e., the number of groups is i); $\hat{C}_{join/leave,i}$, $\hat{C}_{mobility,i}$, $\hat{C}_{beacon,i}$, $\hat{C}_{mp,i}$, and $\hat{C}_{GC,i}$ are the respective cost components, given that the number of groups in the system is i. Note that in the special case that the density is sufficiently high such that nodes are connected all the time, the group partitioning rate is zero, so the merge/partition cost is also zero. Below we explain how we calculate $\hat{C}_{join/leave,i}$, $\hat{C}_{mobility,i}$, $\hat{C}_{beacon,}$, $\hat{C}_{mp,i}$, and $\hat{C}_{GC,i}$.

Basic parameters in the model are summarized in Table 7-2. Table 7-3 gives parameters derived from basic parameters.

7.2.2 Cost for Regional Boundary Cross ($\hat{C}_{mobility}$)

The traffic cost incurred per time unit due to a regional boundary cross event while the system in state i, $\hat{C}_{mobility,i}$, covers two cases: (a) a regional boundary cross by a non-leader; (b) a regional boundary cross by a leader. Thus, $\hat{C}_{mobility,i}$ is given by:

$$\hat{C}_{mobility,i} = \Lambda_m \times \left[C_{mobility}^{non-leader} + C_{mobility}^{leader}\right] \qquad (7\text{-}6)$$

Table 7- 2: Basic Model Parameters.

Parameter	Meaning
σ	Mobility rate per node (*times/s*)
λ	Group join rate per node (*times/s*)
μ	Group leave rate per node (*times/s*)
$\lambda_{np,i}$	Per-group group partition rate when the number of groups = *i* (*times/s*)
$\mu_{nm,i}$	Per-group group merge rate when the number of groups = *i* (*times/s*)
λ_p	Node density (nodes/km^2)
λ_q	Group communication rate per node (*times/s*)
A	Operational area of the mobile group, that is, $A = \pi r^2$ (km^2) where *r* is the radius
A_{region}	Area of a region (km^2)
s	Radius of a hexagon region (*m*)
r	Radius of the operational area (*m*)
R	Wireless per-hop radio range (*m*)
k	Size of a group key (*bits*)
v	Size of each intermediate value in GDH (*bits*)
T_{RB}	Intra-regional beaconing interval (*s*)
T_{LB}	Inter-regional beaconing interval (*s*)
M_{alive}	Size of a beaconing message (*bits*)
M_q	Size of a group communication message (*bits*)
U_{view}	Size of an update message (*bits*)
$C_{np,\, i}$	Cost per group partition event when the number of groups = *i* (*hop bits/s*)
$C_{nm,\, i}$	Cost per group merge event when the number of groups = *i* (*hop bits/s*)
C_{intra}	Cost for intra-key rekeying and regional view updating in a region (*hop bits/s*)
$C_{inter,\, i}$	Cost for inter-key rekeying and leader view updating in a group (*hop bits/s*)
H_{region}	Number of hops between a leader and a member within a region
$H_{leader,\, i}$	Number of hops among leaders in a group when the number of groups = *i*
$R(n)$	Number of regions in the system
$N_{region,\, i}$	Number of regions in a group when the number of groups = *i*
N_{np}^{region}	Number of partitioned regions in a group after a group partition event
N_{nm}^{region}	Number of merged regions in a group after a group merge event
$N_{region}^{members}$	Number of members in a region
C_{rekey}^{intra}	Intra-regional cost for rekeying a regional key (*hop bits/s*)
C_{update}^{intra}	Intra-regional cost for updating a regional view (*hop bits/s*)
$C_{rekey,i}^{inter}$	Inter-regional cost for rekeying a leader key when the number of groups = *i* (*hop bits/s*)
$C_{update,i}^{inter}$	Inter-regional cost for updating a leader view when the number of groups = *i* (*hop bits/s*)
$C_{leader,i}^{change}$	Cost for a leader change in a group when the number of groups = *i* (*hop bits/s*)

Table 7- 3: Derived Parameters.

Parameter	Meaning
$\boldsymbol{\sigma_n}$	Regional mobility rate per node
$\boldsymbol{\Lambda_m}$	Aggregate regional mobility rate
$\boldsymbol{\Lambda_J}$	Aggregate group join rate
$\boldsymbol{\Lambda_L}$	Aggregate group leave rate
$\boldsymbol{\Lambda_{RB}}$	Aggregate periodic beaconing rate from all members in the system
$\boldsymbol{\Lambda_{LB}}$	Aggregate periodic beaconing rate from all leaders in the system
$\boldsymbol{\Lambda_q}$	Aggregate group communication rate from all members in the system

Here Λ_m is the aggregate regional mobility by nodes in the system, given by $\sigma_n \times N$. The cost for the system to handle a non-leader member crossing a regional boundary is given by:

$$C_{mobility}^{non-leader} = P_{non-leader} \times [C_{intra} \times 2] \qquad \text{(7-1)}$$

Here $P_{non\text{-}leader}$ is the probability of a non-leader given by $P_{non\text{-}leader} = (N\text{-}N_{leader})/N$ where N is the total number of nodes and N_{leader} is the number of leaders in the system, C_{intra} is the cost incurred for rekeying K_R and updating the regional view in a region, given in Equation 7-8 below:

$$C_{intra} = \left[C_{update}^{intra} + C_{rekey}^{intra}\right] \times H_{region} \qquad \text{(7-2)}$$

where C_{update}^{intra} is the cost for updating a regional view, C_{rekey}^{intra} is the cost for rekeying a regional key, and H_{region} is the number of hops multicast from a leader to all regional members for updating a regional view or changing key to the members in its region. Note that the number of hops in H_{region} is counted based on the assumption of a balanced binary tree for a multicasting message. Thus, $(N_{region}^{members} - 1)$ is the number of edges where participating nodes are $N_{region}^{members}$ in the balanced binary tree for multicasting. H_{region} is calculated by Equation 7-9:

$$H_{region} = \frac{s}{R} \times \left(N_{region}^{members} - 1\right) \qquad \text{(7-3)}$$

$$s = \sqrt{\frac{2}{3\sqrt{3}} A_{region}} \qquad A_{region} = \frac{A}{R(n)}$$

Here A_{region} is the area of a region, s is the circum-radius of a hexagon-shaped region, and R is the wireless per-hop radio range (m). When there is no region, that is, when $R(n) = 1$ at $n=0$, $A_{region} = A$.

On the other hand, the cost for handling a regional boundary crossing event by a leader member is:

$$C_{mobility\ ,i}^{leader} = P_{leader} \times \left[C_{intra} \times 2 + C_{inter\ ,i} + C_{leader\ ,i}^{change}\right] \tag{7-4}$$

where $C_{inter,\,i}$ is the cost for rekeying a leader key and updating the leader view, given below in Equation 7-11, $P_{leader} = R(n)\ /N$ where $P_{leader} = 1 - P_{non\text{-}leader}$ is the probability of a leader crossing a regional boundary, and $C_{leader\ ,i}^{change}$ is the cost for changing a leader in a region, given below in Equation 7-13.

The cost for inter regional communications ($C_{inter,\,i}$) is computed as:

$$C_{inter\ ,i} = \left[C_{update\ ,i}^{inter} + C_{rekey\ ,i}^{inter}\right] \times H_{leader\ ,i} \tag{7-5}$$

where $C_{update\ ,i}^{inter}$ is the cost for updating the leader view in a group, $C_{rekey\ ,i}^{inter}$ is the cost for rekeying the leader key in a group, and $H_{leader,\,i}$ is the number of hops multicast from one leader to all other leaders in a group where there is i groups. Since there are i groups in the system, the radius of a group can be approximated as $r/\sqrt{i}$ where r is the radius of the operational area. Consequently, the number of hops among leaders in a group, $H_{leader,\,i}$, is given by:

$$H_{leader\ ,i} = \frac{r}{R\sqrt{i}} \times \left(N_{region\ ,i} - 1\right) \quad where\ \ N_{region\ ,i} = R(n)/i \tag{7-6}$$

Similar with calculating the number of hops for H_{region}, the number of hops in $H_{leader,\,i}$ is counted based on the assumption of a balanced binary tree for a multicasting message. Note that ($N_{region,\,i} - 1$) is the number of edges where participating nodes are $N_{region,\,i}$ in the balanced binary tree for multicasting.

For $C_{leader\ ,i}^{change}$, the outgoing leader would multicast two messages announcing its intention to leave, with one message to its regional members using its regional key, and another message to other leaders using a leader key. In addition, the new leader would multicast two messages expressing "I-am-a-new-leader" to its regional members and to leader group using its regional key and the leader key respectively. Further, these messages need to travel through a number of hops at the leader and intra-regional levels represented by $H_{leader,\,i}$ and H_{region}, respectively. Thus, the cost for a leader change, $C_{leader\ ,i}^{change}$, is calculated as:

$$C_{leader,i}^{change} = H_{leader,i} \times \left[M_{old-leader}^{leaders} + M_{new-leader}^{leaders}\right] + \tag{7-7}$$
$$H_{region} \times \left[M_{old-leader}^{regional-members} + M_{new-leader}^{regional-members}\right]$$

Summarizing above, $\hat{C}_{mobility,i}$ is given by:

$$\hat{C}_{mobility,i} = \Lambda_m \times \left\{[2 \times C_{intra}] + P_{leader} \times \left[C_{inter,i} + C_{leader,i}^{change}\right]\right\} \tag{7-8}$$

7.2.3 Cost for Group Join/Leave ($\hat{C}_{join/leave}$)

$\hat{C}_{join/leave,i}$ includes the cost for handling group join and leave. Thus,

$$\hat{C}_{join/leave,i} = \left[\Lambda_J \times C_{join,i}\right] + \left[\Lambda_L \times C_{leave,i}\right] \tag{7-9}$$

Here Λ_J and Λ_L are the aggregate group join and leave rates of all members, respectively, given in Equation 7-1. A group join event requires the update of the regional view and the rekeying of the regional key in the region from which the join event is originated, the cost of which is C_{intra}, as well as the update of the group view and the rekeying of a group key, the cost of which is $C_{group,i}$. $C_{join,i}$ is computed by:

$$C_{join,i} = C_{intra} + C_{group,i} \tag{7-10}$$

where $C_{group,i}$ is given by:

$$\begin{aligned} C_{group,i} &= C_{update,i}^{group} + C_{rekey,i}^{group} \\ &= \left[H_{leader,i} \times M_{update}^{leaders} + H_{region} \times N_{region,i} \right. \\ &\left. \times M_{update}^{regional-members}\right] + \left[H_{region} \times N_{region,i} \times M_{rekey}^{regional-members}\right] \end{aligned} \tag{7-11}$$

Here $M_{update}^{leaders}$ is the number of bits required in a multicast message for updating the group view for the leaders, $M_{update}^{regional-members}$ for updating the group view for members in a region, and $M_{rekey}^{regional-members}$ for rekeying the group key for members in a region; $N_{region,i}$ is the number of regions in a group, given by $R(n)/i$.

The cost for group leave event includes two cases, namely, when a non-leader member leaves and when a leader leaves the group. Thus, the cost for a group leave event is:

$$C_{leave,i} = C_{leave,i}^{non-leader} + C_{leave,i}^{leader} \qquad (7\text{-}12)$$

with

$$C_{leave,i}^{non-leader} = P_{non-leader} \times \left[C_{intra} + C_{group,i}\right] \qquad (7\text{-}13)$$

$$C_{leave,i}^{leader} = P_{leader} \times \left[C_{intra} + C_{inter,i} + C_{group,i} + C_{leader,i}^{change}\right] \qquad (7\text{-}14)$$

where C_{intra}, $C_{inter,\,i}$, $C_{group,\,i}$, and $C_{leader,i}^{change}$ are given earlier in Equations 7-8, 7-11, 7-17 , and 7-13 respectively, and P_{leader} and $P_{non\text{-}leader}$ are as previously described. Here we note that the case in which a leader becomes disconnected, either voluntarily or involuntary, is considered as a leave event whose cost is given by Equation 7-20, accounting for the cost for a leave event by a member (C_{intra} and $C_{group,\,i}$) plus the cost for forming a new leader key ($C_{inter,i}$) and the cost for a leader change ($C_{leader,i}^{change}$).

7.2.4 Cost for Periodic Beacon ($\hat{C}_{beacon}$)

$\hat{C}_{beacon,i}$ includes the cost of beaconing messages in two levels, namely, intra-regional beaconing among members in a region for maintaining the regional view, and inter-regional beaconing among leaders for maintaining the leader view. Thus, $\hat{C}_{beacon,\,i}$ is computed as:

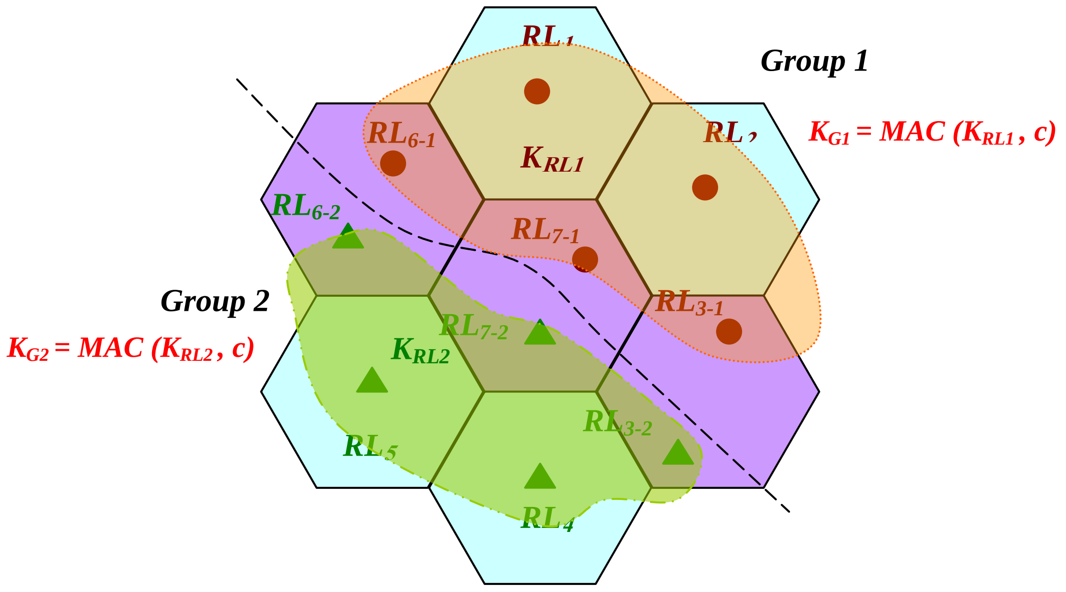
$$\hat{C}_{beacon,i} = \left[\Lambda_{RB} \times M_{alive} \times H_{region}\right] + \left[\Lambda_{LB} \times M_{alive} \times H_{leader,i}\right] \qquad (7\text{-}15)$$

where M_{alive} is the number of bits in a beaconing message and Λ_{RB} and Λ_{LB} are the overall beacon rates in the system by all of its members at the intra-regional level, and by all of its leaders at the inter-regional level, respectively. Λ_{RB} and Λ_{LB} are obtained from the reciprocals of the periodic beaconing intervals, T_{RB} and T_{LB}, at the intra-regional level and at the leader level, respectively, multiplied by the number of members in the operational area, $N\lambda/(\lambda+\mu)$, and the number of leaders (which is the same as the number of regions), $R(n)$, respectively, i.e.,

$$\Lambda_{RB} = N \times \left[\frac{\lambda}{\lambda+\mu}\right] \times \frac{1}{T_{RB}} \qquad (7\text{-}16)$$

7.2.5 Cost for Group Merge/Partition ($\hat{C}_{mp,\,i}$)

The traffic cost incurred per time unit due to group partition/merge while the system is in state i, $\hat{C}_{mp,i}$, is the sum of that due to group partition, $C_{partition,i}$, and that due to group merge, $C_{merge,i}$. We observe that $C_{merge,\,1} = 0$ since in state 1 there is no merge event.



🟧 ***For Group 1, RL_1, RL_2, $RL_{3\text{-}1}$, $RL_{6\text{-}1}$ and $RL_{7\text{-}1}$ are engaged in contributory key agreement to generate K_{RL1} and K_{G1}.***

🟩 ***For Group 2, RL_4, RL_5, $RL_{3\text{-}2}$, $RL_{6\text{-}2}$ and $RL_{7\text{-}2}$ are engaged in contributory key agreement to generate K_{RL2} and K_{G2}.***

⬢ ***Regions 3, 6, and 7 are "affected" regions, each being partitioned into two regions such that a regional key is rekeyed in each partitioned region.***

Figure 7-6: Dynamic Reconfiguration in response to Group Partition under *R(1)* = 7.

Figure 7-6 illustrates a group partition event for the case in which the number of hexagon regions is 7. Initially the system contains one group. Later the group is split into two groups, 1 and 2. Regions 3, 6, and 7 are the ones on which group partition occurs. As a result, regions 3, 6, and 7 are each partitioned into two regions, e.g., region 3 is partitioned into region 3-1 and region 3-2, and region 6 is partitioned into regions 6-1 and 6-2, and so on. In each of these "partitioned" regions, a regional key needs to be rekeyed since the membership in the region has been changed. On the other hand, regions 1 and 2 in group 1 as well as regions 4 and 5 in group 2 are not affected by the group partition event, so the regional key needs not be rekeyed in these regions. After group partition, group 1 contains 5 regions, labeled as RL_1, RL_2, $RL_{3\text{-}1}$, $RL_{6\text{-}1}$ and $RL_{7\text{-}1}$, while group 2 also contains 5 regions, labeled as RL_4, RL_5, $RL_{3\text{-}2}$, $RL_{6\text{-}2}$ and $RL_{7\text{-}2}$. In group

1, RL_1, RL_2, $RL_{3\text{-}1}$, $RL_{6\text{-}1}$ and $RL_{7\text{-}1}$ then execute CKA to generate K_{RL1} and subsequently $K_{G1},$ while in group 2, RL_4, RL_5, $RL_{3\text{-}2}$, $RL_{6\text{-}2}$ and $RL_{7\text{-}2}$ are engaged in CKA to generate K_{RL2} and subsequently $K_{G2.}$

We calculate $C_{partition,i}$ as the product of the group partitioning rate at state i, $\lambda_{np\ i}$, and the cost per group partition event in state i, $C_{np,i}$, i.e.,

$$C_{partition\ ,i} = \lambda_{np,i} \times C_{np,i} \qquad (7\text{-}17)$$

$$C_{np,i} = 2 \times \left[\left(N_{np}^{region} \times C_{intra}\right) + C_{inter\ ,i} + C_{group\ ,i}\right] + N_{np}^{region} \times C_{leader\ ,i}^{change} \qquad (7\text{-}18)$$

Here $C_{np,i}$ covers four costs: an intra-regional cost for rekeying a regional key in each of the "partitioned" regions in a group ($\left(N_{np}^{region} \times C_{intra}\right)$), an inter-regional cost for rekeying a leader key in each partitioned group ($C_{inter\ ,i}$), a cost for rekeying a group key in each partitioned group ($C_{group,\ i}$), and a cost for changing leader in each partitioned group ($C_{leader\ ,i}^{change}$). Since a group partition event results in two partitioned groups, the incurred cost is multiplied by two. Note that C_{intra}, $C_{inter,\ i}$, $C_{group,\ i}$, and $C_{leader\ ,i}^{change}$are per-group costs at state i, given by Equation 7-8, 7-11, 7-17, and 7-13 respectively. Also $\lambda_{np,i}$ is the number of "partitioned" regions in a group after group partition; it is equal to one if the group size is smaller than the regional size; otherwise it is equal to the ratio of the group size to the regional size, viz.,

$$\begin{aligned} & if\left(1/\sqrt{i} > s\right) \qquad (7\text{-}19) \\ & \quad N_{np,i}^{region} = \frac{r}{s\sqrt{i}}; \\ & else \\ & \quad N_{np,i}^{region} = 1; \end{aligned}$$

where $r/\sqrt{i}$ is the radius of a group after group partition and s is the radius of a hexagonal region.

Next we compute the traffic cost incurred per time unit due to group merge while the system in state i, $C_{merge,,i}$; it is computed by the product of the group merging rate at state i, $\mu_{nm,i}$, and the cost per group merge event in state i, $C_{nm,i}$, i.e.,

$$C_{merge\ ,i} = \mu_{nm\ ,i} \times C_{nm\ ,i} \qquad (7\text{-}20)$$

$$C_{nm,i} = \left(N_{nm,i}^{region} \times C_{intra}\right) + C_{inter,i} + C_{group,i} + C_{leader,i}^{change} \qquad (7\text{-}21)$$

Here the cost per group merge event in state i ($C_{nm,i}$) covers 4 cost components: an intra-regional cost in the merged group $\left(N_{nm,i}^{region} \times C_{intra}\right)$, an inter-regional cost in the merged group ($C_{inter,\,i}$), a cost for rekeying a group key and updating a group view in the merged group ($C_{group,\,i}$), and a cost for changing leader in the merged group ($C_{leader,i}^{change}$). Similar to the calculation of $N_{np,i}^{region}$, $N_{nm,i}^{region}$ is calculated based on Equation 7-25 except that the radius of the group after group merge is being used in the calculation.

Summarizing above, the cost per time unit due to group partition/merge events while the system in state i, $\hat{C}_{mp,i}$, is the sum of that due to group partition and that due to group merge:

$$\hat{C}_{mp,i} = C_{partition,i} + C_{merge,i} \qquad (7\text{-}22)$$

7.2.6 Cost for Group Communications ($\hat{C}_{GC}$)

$\hat{C}_{GC,\,i}$ includes the cost of group communications between members. We assume that each node communicates with each other with the rate of λ_q. That is, each node requests or publishes data to all group members with the time interval of $1/\lambda_q$. Thus, the aggregate rate that data are published in each node is obtained as:

$$\Lambda_q = N \times \left[\frac{\lambda}{\lambda + \mu}\right] \times \lambda_q \qquad (7\text{-}23)$$

Whenever each node publishes its data, thus published data should be disseminated to all members. Taking advantage of our hierarchical key management structure, the published data can be distributed to all leaders first, and then each leader can multicast them to its members in the region.

$$\hat{C}_{GC,i} = \Lambda_q \times \left[\left(N_{region,i} \times M_q \times H_{region}\right) + \left(M_q \times H_{leader,i}\right)\right] \qquad (7\text{-}24)$$

7.2.7 Parameterization

Without loss of generality, we exemplify our region-based key agreement protocol with GDH [92] as the key agreement protocol to be used for secret key generation. We use it at both the intra-regional and inter-regional (leader) levels. GDH.3 is already explained in Section 3.2.2.

Below we briefly explain how we parameterize C_{rekey}^{intra} and C_{rekey}^{inter} as a function of the number of participants, N based on GDH.3. The cost is measured by the number of information bits required by GDH.3 multiplied by the number of hops information bits are transmitted. We apply GDH.3 as the key agreement protocol at both the intra-regional and inter-regional levels. In the former case, N is the number of members in a region; in the latter case, N is the number of leaders in a group.

In [83], the secrecy property of GDH has been proven. Since it is possible that join/leave operations may occur frequently in our target applications, we adopt an optimized way discussed in [91] with some modification to perform join/leave operations. More specifically, where there is N members in the system, M_N saves the contents of the original multicast and response messages from stages 2 and 3 in Figure 3-2. When a new member, say, M_{N+1}, joins, it forwards its contribution to M_N. Then, M_N uses the new contribution along with contributions of existing members saved to generate a new set of subkeys, and distributes it to all members as described in stage 4 of Figure 3-2 (Chapter 3). Therefore, a new member join process only requires two additional rounds. The *backward secrecy* is preserved because a new exponent is used in generating a new set of subkeys by M_N. For a member leave, if M_p is a member to be removed from the group (i.e., a member leave), M_N takes the special role of generating a new set of $N – 2$ subkeys (excluding a portion of leaving member, M_p) by using a new exponent. Then, M_N distributes the new set of subkeys to all members. Finally, each member can generate a new group key using the new set of subkeys. Since the subkey for M_p is not included in the received subkeys, M_p cannot generate a new group key, thus preserving the *forward secrecy*. A member leave process only needs one round. If M_N is the member to be removed from the group, M_{N-1} takes the special role of distributing a set of subkeys. These optimized join and leave operations are mainly developed to reduce the number of rounds where bandwidth is a crucial issue. We use these optimized join/leave operations only if a leader is not involved in a join/leave operation.

$Stage\ 1: unicast$

$$Number\ of\ hops\ \times message\ size = 1 \times v\left(N_{region}^{members}\ - 2\right)$$

$Stage\ 2: broadcast$

$$Number\ of\ hops\ \times message\ size = H_{region}\ \times v$$

$Stage\ 3: unicast$

$$Number\ of\ hops\ \times message\ size = H_{region}\ \times v\left(N_{region}^{members}\ - 1\right)$$

$Stage\ 4: broadcast$

$$Number\ of\ hops\ \times message\ size = H_{region}\ \times v\left(N_{region}^{members}\ - 1\right)$$

Figure 7-7: Parameterizing Intra-Regional Communication Cost *(C_{rekey}^{intra})* based on GDH.

Figure 7-7 shows how C_{rekey}^{intra} in the unit of *hop-bits* for rekeying a regional key is calculated after considering the number of hops information bits travel in each step of GDH. Here $N_{region}^{members}$ refers to the number of members in a region and *H_{region}* is calculated by Equation 7-9. C_{rekey}^{intra} is calculated as the total cost from these four stages. The cost of stage 1 is the number of bits required in stage 1, *v* ($N_{region}^{members}$ – 2) as shown in Figure 7-8. Here the number of hops is 1 because in stage 1 of GDH, the message exchange is transmitted in a series fashion from M_1 to M_{N-1} and each node can communicate its neighbor with only one hop[1]. The cost of stage 2 is computed by the number of bits required for stage 2, namely, *v*, multiplied with the average number of hops in a region, H_{region}. The cost of stage 3 is the number of bits required, *v* ($N_{region}^{members}$ –1), multiplied with the number of hops these bits travel, H_{region}. Here the number of hops is H_{region} because in stage 3 of GDH each member sends an intermediate value (*v*) to a regional leader. The cost of stage 4 is computed by the number of bits required for stage 4, namely, *v* ($N_{region}^{members}$ –1), multiplied with the average number of hops in a region, H_{region}. Note that the cost for stage 3 is slightly overestimated by using the maximum number of hops that a message may travel.

[1] In stage 1, the chain can be dynamically formed by having each node pick the nearest node that has not been put on the chain to unicast the next message to. By this way two consecutive nodes on the chain would be mostly one-hop apart except for the last few nodes. Nevertheless, the network traffic cost for stage 1 is underestimated because the last few nodes on the chain could be multi-hop apart.

$Stage\ 1: unicast$

$$Number\ of\ hops\ \times message\ size = H_{leade\ r}^{stage\ 1(GDH)} \times v(N_{region\ ,i} - 2)$$

$Stage\ 2: broadcast$

$$Number\ of\ hops\ \times message\ size = H_{leader\ ,i} \times v$$

$Stage\ 3: unicast$

$$Number\ of\ hops\ \times message\ size = H_{leader\ ,i}^{stage\ 3(GDH)} \times v$$

$Stage\ 4: broadcast$

$$Number\ of\ hops\ \times message\ size = H_{leader\ ,i} \times v(N_{region\ ,i} - 1)$$

Figure 7-8: Parameterizing Inter-Regional Communication Cost *($C_{rekey,i}^{inter}$)* based on GDH.

Figure 7-8 shows how $C_{rekey\ ,i}^{inter}$ for rekeying the leader key is computed. The computational steps are similar to those in Figure 7-7. The number of participants (*N*) now is the number of leaders in a group which is the same as the number of regions in the group, $N_{region,\ i}$ = *R (n)/i*. The average number of hops between two leaders, $H_{leader,\ i}$, is obtained based on Equation 7-12 and is used in stages 2 and 4. The number of hops information bits travel between two leaders in stage 1 is given by:

$$H_{ledaer}^{stage\ 1(GDH)} = \frac{2s}{R} \qquad (7\text{-}25)$$

where *s* is the radius of a region calculated by Equation 7-9 and *R* is the per-hop wireless radio range. This computation is based on the assumption that the leader is located at the center of each region. Thus, the distance between leaders in one ring level (*n*) of a hexagonal area and those in another is simply the diameter of a region. On the other hand, the number of hops information bits travel in stage 3 is given by:

$$H_{leader\ ,i}^{stage\ 3(GDH)} = \left(\sum_{k=1}^{n} 6k \left\lceil \frac{2s}{R} \right\rceil \right) / i \qquad (7\text{-}26)$$

where *n* is the level of ring in the hexagonal area, *s* is the radius of a region, *i* is the number of groups observed, and *R* is the wireless radio range. The distance between a leader and another leader is approximately calculated as the diameter of a hexagon region. Equation 7-32 means adding up all the number of hops by leaders in each level divided by the number of groups observed. In stage 3 of Figure 7-8, since Equation 7-32 already includes $N_{region,\ i} - 1$ transmissions, the unicast cost is simply calculated by multiplying $H_{leader\ ,i}^{stage\ 3(GDH)}$ with *v*, the intermediate message size.

Next we explain how we parameterize the per-group merging/partitioning rates at state *i*. While an analytical model is possible, we opt to run a simulation study to better reflect the dependency of the per-group merging/partitioning rates with node mobility and node density. We collect the number of merge and partition events observed during a sufficiently long period of time *T*. We also sum the sojourn time in which *i* groups are observed in the system. Let S_i denotes this sojourn time that the system stays in state *i* (in which the number of groups = *i*). Let $N_{nm,i}$ and $N_{np,i}$ be the numbers of merge or partition events observed at the number of groups = *i*, respectively. Then, the merging/partitioning rates when the number of groups observed is *i*, represented by $\mu_{nm,i}$ and $\lambda_{np,i}$, are given by:

$$\mu_{nm\ ,i} = \frac{N_{nm\ ,i}}{S_i} \qquad \mu_{np\ ,i} = \frac{N_{np\ ,i}}{S_i} \tag{7-27}$$

We use the measured $\mu_{nm,i}$ and $\lambda_{np,i}$ obtained above to parameterize the *birth-death* process model shown in Figure 7-5.

Figure 7-9 shows the merging/partitioning rates collected from simulation as a function of regional mobility rate (σ) when the node density (λ_p) = 150, per-hop wireless radio range (*R*) = 200 *m*, and initial operational area (*A*) = $\pi\ r^2$ where *r* = 1 *km* with other default parameters listed in Table 7-4. Under these conditions, we observed only a maximum of 5 groups in simulation.

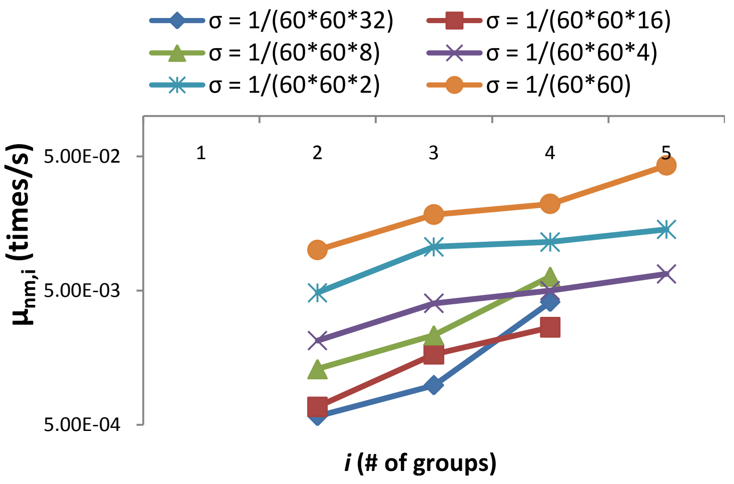


Figure 7-9: (a) Group Merging Rate ($\mu_{nm,i}$).

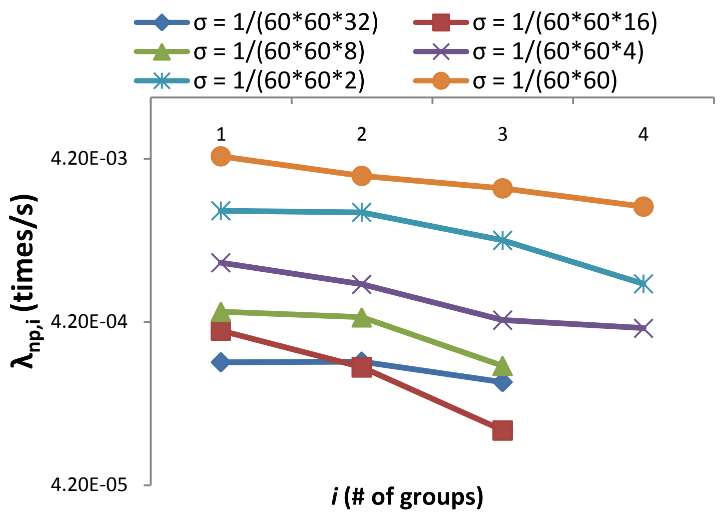


Figure 7-9: (b) Group Partitioning Rate ($\lambda_{np,i}$).

The general tendency observed here is that as the number of groups increases, the group partitioning rate decreases and the group merging rate increases. As shown in Figure 7-9, the general trends show that as the regional mobility rate (σ) increases, higher per-group merging/partitioning rates are observed with the merging rate having a higher sensitivity over the partitioning rate. The trend also shows that as the regional mobility rate decreases, a group is less likely to be partitioned but groups are more likely to be merged, and vice versa.

7.3 Numerical Results

7.3.1 Numerical Analysis

Below we report numerical results for the communication cost incurred per time unit in executing the adopted region-based group key management protocol as a function of model parameters. We demonstrate that there exists an optimal regional size that minimizes the overall communication cost. The effect of regional size is represented by a parameter, n, where $n = 0$ means that there is only one region (our baseline model), $n = 1$ means 7 regions, $n = 2$ means 19 regions, and so on. Note that this parameter n computes the total number of regions that exist in the entire operational area (i.e., $R(n) = 3n^2 + 3n + 1$).

We evaluate the effect of the number of regions in the system on the overall cost ($\hat{C}_{total}$) given in Equation 7-4 while varying other critical parameters (i.e., regional mobility rate σ, node density λ_p, and group communication rate λ_q) to test their effects. By means of graphs, we demonstrate that the optimal regional area size can be changed dynamically in response to changing environments and network conditions characterized by a set of model parameter values. Table 7-4 shows the default parameter values used in this case study. The wireless radio range (*R*) is set at 200 *m*, as in IEEE 802.11, under which we observe a reasonable number of groups in response to group partition or merge events. The key size for the group key (*k*) or the intermediate key (*v*) is chosen to be 64 bits as commonly used for cryptographic keys. The beaconing intervals for the beaconing messages by regional members and by leader members (T_{RB} and T_{LB}, respectively) are chosen to be 5 *s* and 2 *s*, respectively, to allow the system to quickly respond to node disconnection/failure events. The shorter interval is used for leader beaconing because of the importance of leader role. The size of a beaconing message (B_{alive}) is chosen to be only 32 *bits* since it is just a simple "I-am-alive" message. Finally, the size of a view update message (U_{view}) is chosen to be 500 *bits* so it is sufficiently large to contain information such as member identifiers and their locations.

Table 7-4: Default Parameter Values.

Parameter	Default Value
σ	1/(60*60*32)
λ	1/(60*60)
μ	1/(60*60*4)
λ_q	1/(60*10)
λ_p	150 *nodes/km*^{2}
A	π *km*^{2}
R	200 *m*
k	64 *bits*
v	64 *bits*
T_{RB}	5 *s*
T_{LB}	2 *s*
U_{view}	500 *bits*
M_{alive}	32 *bits*
M_q	500 *bits*

Figure 7-10 shows optimal regional sizes versus mobility rate (σ). More specifically, Figure 7-10 (a) shows the impact of *n* on the total cost ($\hat{C}_{total}$) when varying σ, while Figure 7-10 (b) compares $\hat{C}_{total}$ between the baseline system (i.e., non-region-based group key management) versus the system at the "optimal regional size" (i.e., the optimal regional size, 37 regions here, is used under region-based group key management). We see that as the regional size decreases (as *n* increases), $\hat{C}_{total}$ decreases until it reaches the optimal point at N_{region} = 19 that would minimize $\hat{C}_{total}$, after which $\hat{C}_{total}$ increases again beyond that point.

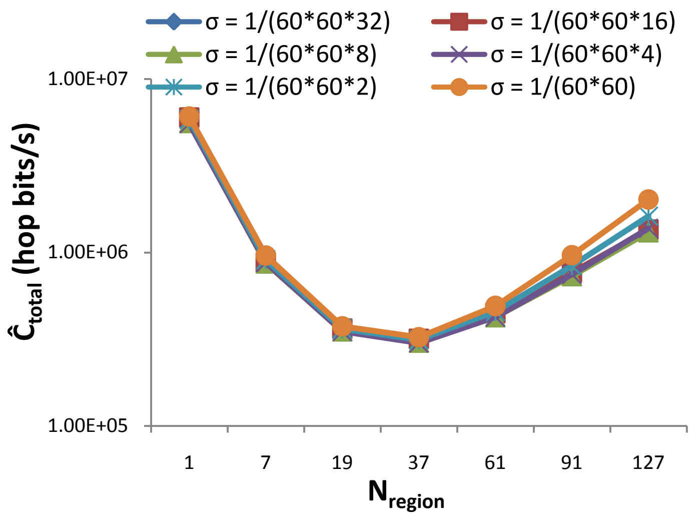


Figure 7-10: (a) Overall Cost ($\hat{C}_{total}$) versus Number of Regions (N_{region}) as a Function of Mobility Rate (σ).

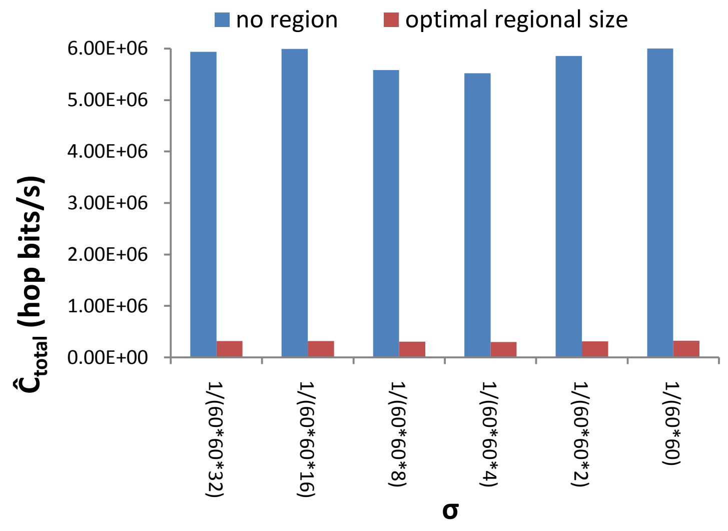


Figure 7-10: (b) Overall Cost ($\hat{C}_{total}$) Under No Region versus Optimal Regional Size as a Function of Mobility Rate (σ).

Note that a higher N_{region} indicates there are fewer members in a region. The reason that an optimal N_{region} exists is that as N_{region} increases, the inter-regional overhead (i.e., updating and rekeying cost at a leader level) increases, while the intra-regional overhead (i.e., updating and rekeying cost at a regional level) decreases. Initially, the total communication cost decreases as the number of regions increases because of the decreasing intra-regional overhead while it increases again after the optimal N_{region} reaches because of the increasing inter-regional overhead. As shown in Figure 7-10 (b), the network traffic generated under the optimal regional size is significantly lower than that under the no-region protocol.

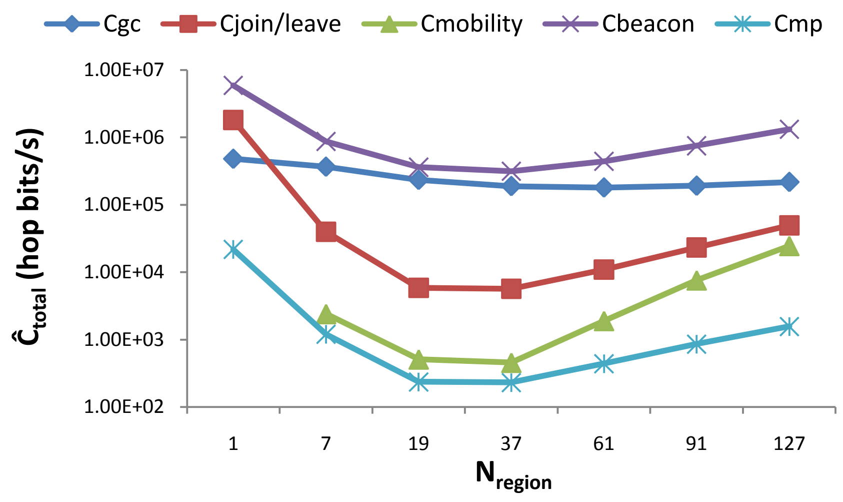


Figure 7-11: (a) Breakdown of $\hat{C}_{GC}$, $\hat{C}_{mobility}$, $\hat{C}_{join/leave}$, $\hat{C}_{beacon}$, and $\hat{C}_{mp}$ versus Number of Regions (N_{region}).

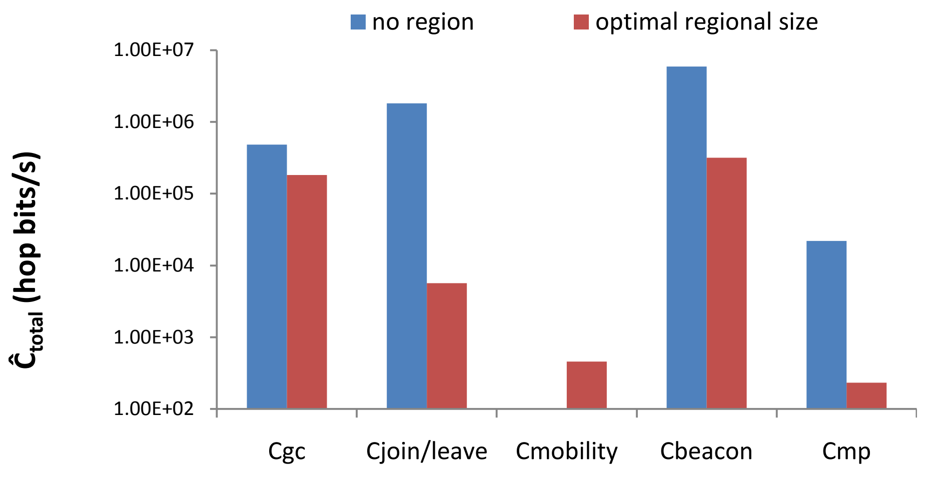


Figure 7-11: (b) $\hat{C}_{GC}$, $\hat{C}_{mobility}$, $\hat{C}_{join/leave}$, $\hat{C}_{beacon}$, and $\hat{C}_{mp}$ under No Region versus Optimal Regional Size.

Figure 7-11(a) breaks down the overall cost ($\hat{C}_{total}$) into its constituents $\hat{C}_{GC}$, $\hat{C}_{mobility}$, $\hat{C}_{join/leave}$, $\hat{C}_{beacon}$, and $\hat{C}_{mp}$ as a function of N_{region} with all default values used. We use this case to explain how an optimal $N_{region} = 37$ for the overall communication cost ($\hat{C}_{total}$) is obtained. The five cost components offset each other, and then the optimal point of overall communication cost ($\hat{C}_{total}$) is determined at $N_{region} = 37$ by trading the intra-regional cost off for the inter-regional cost. As illustrated in Figure 7-11 (b), while the "no region" protocol does not have $\hat{C}_{mobility}$ and small

$\hat{C}_{GC}$, our protocol at the "optimal regional size" has small $\hat{C}_{mobility}$. However, by having significant cost savings in three other cost components ($\hat{C}_{beacon}$, $\hat{C}_{join/leave}$, $\hat{C}_{mp}$) at the optimal regional size, our protocol substantially improves the system performance.

We also observed in both Figure 7-11 (a) and 7-11(b) that $\hat{C}_{join/leave}$ and $\hat{C}_{merge/partition}$ shows striking cost savings compared to other communication cost occurred. Note that group communication cost, $\hat{C}_{GC}$, does not have much benefit of using hierarchical group key management structure showing not significantly sharp curve even though it has also an optimum at N_{region} = 61 as shown in Figure 7-11 (a).

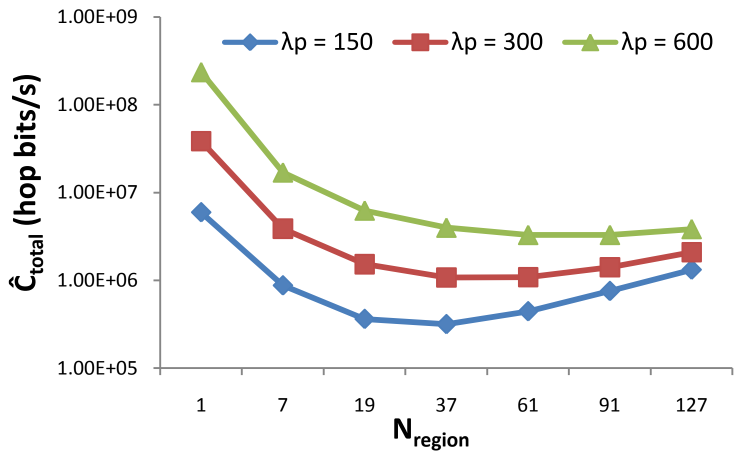


Figure 7-12: (a) Overall Cost ($\hat{C}_{total}$) versus Number of Regions (N_{region}) as a Function of Node Density (λ_p).

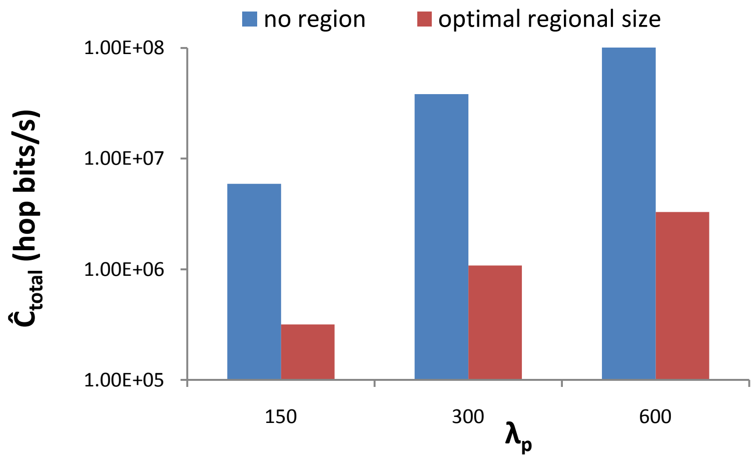


Figure 7-12: (b) Overall Cost ($\hat{C}_{total}$) under No Region versus Optimal Regional Size as a Function of Node Density (λ_p).

Figure 7-12 (a) shows the effect of node densities λ_p on $\hat{C}_{total}$. This case study varies λ_p from 150, 300, to 600. First, as λ_p increases, $\hat{C}_{total}$ increases because of the increased number of members in each region. Thus, increasing λ_p introduces a higher intra-regional cost for updating the regional membership information and rekeying the regional key. Here it should be noted that since the number of leaders remains the same under different λ_p, $\hat{C}_{total}$ increases as λ_p increases only due to an increased intra-regional overhead. Second, we observe that the optimal N_{region} shifts to the right as λ_p increases. That is, a smaller λ_p produces a smaller optimal N_{region} (e.g., optimal N_{region} = 37 at λ_p = 150 and 300) while a larger λ_p generates a larger optimal N_{region} (e.g., optimal N_{region} = 61 at λ_p = 600). This happens because the intra-regional cost always favors placing fewer members in a region and thus a smaller regional size is favored under high λ_p. Conversely, a large regional size is preferred under low λ_p. Lastly, we notice that as N_{region} increases, $\hat{C}_{total}$ converges to almost the same point, e.g., $\hat{C}_{total}$ at N_{region} = 127. The reason is that in the extreme case where there are many regions, there is little intra-regional overhead and the inter-regional overhead dominates, thus causing $\hat{C}_{total}$ converged to the same value. Figure 7-12 (b) again shows that our protocol operating at the optimal regional size produces significant cost saving compared with the non-region-based group key management protocol. Noticeably, as λ_p increases, the cost saving is more pronounced.

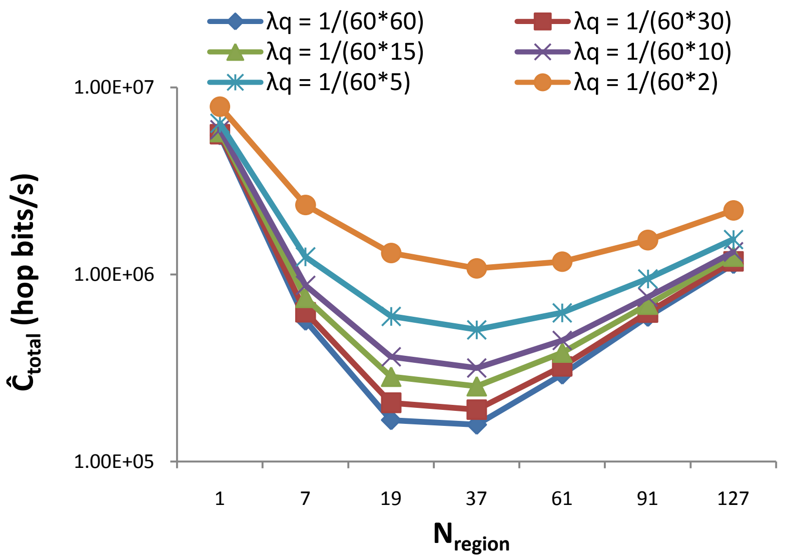


Figure 7-13: (a) Overall Cost ($\hat{C}_{total}$) versus Number of Regions (N_{region}) as a Function of Group Communication Rate (λ_q).

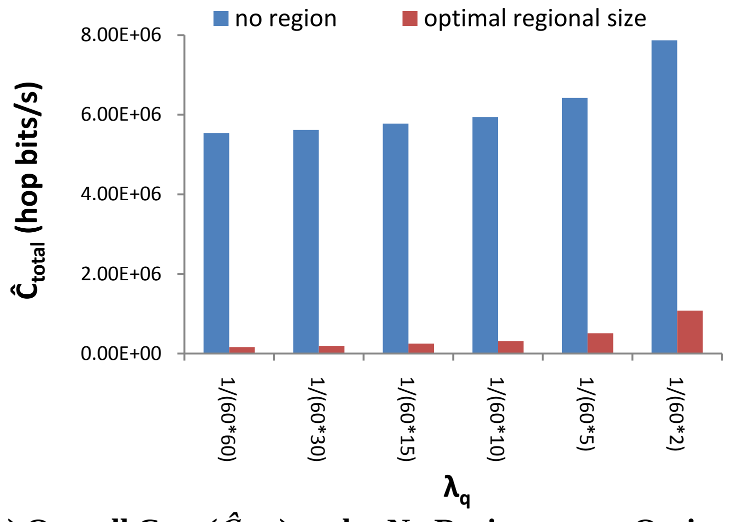


Figure 7-13: (b) Overall Cost ($\hat{C}_{total}$) under No Region versus Optimal regional size as a Function of Group Communication Rate (λ_q).

Finally, Figure 7-13(a) shows the effect of group communication rate, λ_q, on the overall communication cost ($\hat{C}_{total}$). In all cases, an optimal N_{region} is found at 37. Notice that an optimal N_{region} = 37 is identified with more distinct difference from $\hat{C}_{total}$ at N_{region} = 61 as λ_q becomes low. On the other hand, as λ_q becomes high, an optimal N_{region} identified has less distinct difference from $\hat{C}_{total}$ at N_{region} = 61. This phenomenon is due to the fact that group communication cost ($\hat{C}_{GC}$) does not have much benefit of using a hierarchical key management structure. Figure 7-13(b) shows that our region-based group key management protocol at optimizing setting with N_{region} = 37 performs much better than the no-region counterpart in the total traffic generated ($\hat{C}_{total}$), while still satisfying the secrecy and survivability requirements of the system.

7.3.2 Simulation Results

We have conducted a simulation study to validate analytical results. The simulation program is implemented based on a discrete-event simulation language called *SMPL* [90].

We populate the MANET area based on a selected node density and the default parameter values listed in Table 7-4. For example, when λ_p = 150, we randomly place 150 nodes per km^2, thus having a total of 480 nodes approximately in the π km^2 operational area in our simulation. In this case, the initial number of member nodes, i.e., $N\,\lambda/(\lambda+\mu)$, is 392 and they are scattered in the operational area. All nodes can be connected through multiple hops using a per-hop wireless radio range = 200 *m*. Multiple groups may be observed in the operational area. Each node in its lifecycle could generate six events, namely, GROUP JOIN, GROUP LEAVE, BEACON, MOBILITY (i.e., a node's regional boundary crossing), GROUP COMMUNICATION, GROUP MERGE, and GROUP PARTITION. For the MOBILITY event, we use *random waypoint mobility (RWM)* to model the movement of a node with mobility rate of σ following Equation 5-27. For MOBILITY events generated by a node, the time at which a boundary crossing will occur is calculated based on the current speed and direction of the node, and then a MOBILITY event is scheduled accordingly. For GROUP JOIN and GROUP LEAVE events, we assume the inter-arrival times are exponentially distributed with the rates of λ and μ respectively. Thus, GROUP JOIN and GROUP LEAVE events are scheduled based on random values generated by *expntl* $(1/\lambda)$ and *expntl* $(1/\mu)$. Lastly, GROUP COMMUNICATION and BEACON events are scheduled periodically.

The whole MANET area is divided into equal-sized hexagons based on the hexagonal network coverage model with parameter n. Nodes are confined within the MANET area but otherwise can move freely within that area based on the *RWM* model. The simulation keeps track of the location of each node. Thus, at any given time, it knows which region a member belongs to. Whenever an event that affects a regional view occurs, the regional view information is updated. The simulation is event-driven and the cost associated with each event is calculated based on the knowledge of exact locations of nodes in calculating the number of hops, i.e., H_{region} and $H_{leader,\ i}$. We used a technique called *batch mean analysis* [81] to assure the statistical significance of our simulation results in terms of the cost measurements collected (each called an *observation*). The simulation period consists of batches with each batch consisting of 200,000 observations. Tests are performed after running a minimum of 10 batches to make sure that the number of observations is large enough so that the batch means are approximately independently and normally distributed. Then we calculate the grand mean out of batch means collected such that the grand mean is from the true mean by a *confidence interval (CI)* of 95% and an *accuracy level* of 10%. Further, in order to remove the initial transient (warm-up) problem, the first batch discards the first 200 sample values (observations). We discard only 200 observations because the histogram of the first batch does not show much fluctuation after the first 200 observations are discarded as opposed to the first 2000 observations being discarded.

To compare results obtained from different system configurations, the *variance reduction technique (VRT)* [90] of *common random numbers (CRN)* has been used in this case study. Default parameter values as listed in Table 7-4 are used to set up the simulation environment.

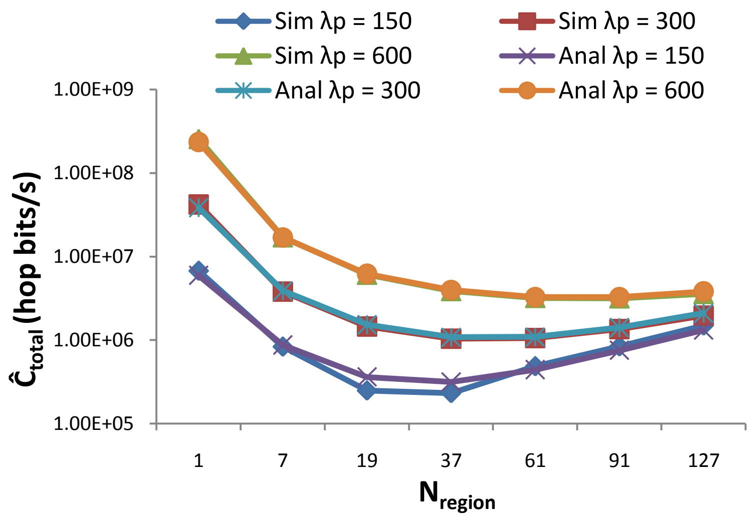


Figure 7-14: Analytical versus Simulation Results: Overall Cost ($\hat{C}_{total}$) versus Number of Regions (N_{region}) as a Function of Node Density (λ_p).

Figure 7-14 shows our simulation results firmly validate our analytical results. The simulation results are obtained from our simulation under the same conditions of Figure 7-12. These curves look remarkably similar to those obtained analytically exhibiting the same trend despite the fact that one set is calculated analytically while the other set is obtained through simulation by averaging the statistical data collected through *batch mean analysis (BMA)*. The standard errors and mean percentage differences between simulation and analytical results are (382, 2%), (1352, 6%), and (4418, 6%) when the node density (λ_p) is at 150, 300, and 600, respectively. The slight difference between analytical results and simulation results is due to the way we calculate the number of hops for computing intra-regional and inter-regional costs and the way we use *RWM* to approximate the per-node mobility rate. In the simulation, we keep track of the location of each node and use exact locations of nodes in the system to calculate the number of hops separating any two nodes when calculating the average C_{rekey}^{intra} and C_{rekey}^{inter}, while in the analysis, we use Equations 7-8, 7-9, 7-11, and 7-12 to compute the number of hops. Nevertheless, the average cost obtained from simulation along with the trend exhibited as the number of regions increases is remarkably close to that obtained analytically, showing the similar optimal regional size to minimize the overall communication cost.

7.4 Summary

In this chapter, we have proposed and analyzed a scalable and efficient region-based secure group key management protocol to support secure group communications in MANETs. The region-based group key management protocol proposed not only reduces network communication costs, but also preserves secrecy and survivability security properties. By using GDH as an exemplary key agreement protocol for group key generation and rekeying at the intra-regional and inter-regional (leader) levels, we discovered that there exists an optimal regional size that would minimize the overall network traffic, when given a set of parameter values characterizing the operational condition. The existence of the optimal regional size is due to a tradeoff between inter-regional and intra-regional overheads.

The scalable region-based group key management protocol developed can be investigated QoS-aware IDS protocols (discussed in Chapter 5) into a protocol suite for supporting QoS-aware mobile group communications in multi-hop MANETs to deal with both outsider and insider attacks. This subject will be treated in Chapter 8. Because nodes can fail and compromised nodes detected by IDS may be evicted permanently, it requires a revisit of the definition of security/performance metrics as time-averaged quantities in order to properly evaluate the integrated protocol suite, including *MTTSF*, network traffic and delay and service response time. This integration also involves a design modification of the IDS protocol to leverage the region-based key management structure with dynamic regional membership information such that neighboring nodes close by, e.g., within a region, may participate in voting to evict a target neighbor node for scalability and dynamic reconfigurability.

Chapter 8 INTRUSION DETECTION INTEGRATED WITH REGION-BASED GROUP KEY MANAGEMENT

In this chapter, we integrate intrusion detection protocols for dealing with insider attacks with hierarchical group key management for dealing with outsider attacks and scalability in MANETs. We perform a tradeoff analysis between security and performance of the system, and identify optimal settings in terms of the optimal intrusion detection interval for IDS and regional area size for region-based group key management under which the system *MTTSF* is maximized while the total group communication cost incurred is minimized in GCSs, when given a set of parameters characterizing the operational and environmental conditions of the GCS. The content is largely based on our published/submitted work in [23, 28].

The rest of this chapter is organized as follows. Section 8.1 gives a detailed description of integrating IDS with region-based group key management. Section 8.2 develops a performance model to assess the performance and security properties of the system. Section 8.3 presents numerical results obtained demonstrating the effectiveness of the design compared with a baseline system without region-based group key management. Section 8.4 summarizes this chapter.

8.1 Integration of Intrusion Detection with Region-based Group Key Management for QoS-aware GCSs

We integrate QoS-aware intrusion detection protocols proposed in Chapter 5 to deal with inside attackers with region-based group key management described in Chapter 7 to deal with outside attackers. The system model follows that for Chapters 5 and 7. The MANET environment with no central key server is considered. Below Section 8.1.1 explains the specifics of how IDS is integrated with region-based hierarchical key management. Section 8.1.2 describes new events resulting from the integration and how the system deals with these events.

8.1.1 IDS Protocols

In integrating IDS with region-based group key management, we consider two voting-based IDS schemes for GCSs in multi-hop MANETs: *region-voting-based IDS* and *group-voting-based IDS*. The two voting-based schemes are differentiated by the way *m* vote-participants are selected when evaluating a target node. Each node periodically exchanges its routing information, location, and *id* to its neighboring nodes. In region-voting-based IDS, only nodes in the same geographical region are candidates as vote-participants with respect to a target node. In group-voting-based IDS, all nodes in the group are candidates as vote-participants against a target node. The selection of *m* vote-participants from member nodes in a region or in the group affects the security and performance aspects of voting-based IDS, which we aim to study its effect. In either case, a coordinator would be elected based on an election protocol among candidate nodes. The coordinator node will select *m* nodes randomly (including itself), and multicasts this list of *m* selected nodes (serving as vote-participants against a target node) to all group members. Each vote-participant node selected will independently vote for or against the target node by disseminating its vote to all group members. Vote authenticity is achieved via preloaded public/private key pairs. By this way, all group members know who *m* vote-participants are, and, based on votes received, can determine whether or not a target node is considered compromised and needs to be evicted for security reasons.

8.1.2 Region-based Group Key Management

In addition to GROUP JOIN/LEAVE, MOBILITY, BEACON, GROUP MERGE/ PARTITION, and GROUP COMMUNICATION events, the integration of IDS with region-based group key management introduces two more events as follows:

EVICTION: After a node is detected as compromised, the group key K_G is rekeying based on GDH to evict the compromised node.

INTRUSION DETECTION: Messages required for IDS activities follow the rules for group communication, including status exchange, vote-participant selection, vote-participant-list dissemination and vote dissemination. A target node is examined by IDS periodically and if the target node is considered compromised, it will be evicted by rekeying the group key K_G based on GDH.

8.2 Performance Analysis

8.2.1 Performance Model

We again use *MTTSF* and $\hat{C}_{total}$ as metrics to measure security and performance properties of GCSs in MANETs. The $\hat{C}_{total}$ metric indicating the total traffic incurred per time unit (*s*) now also includes traffic introduced due to the two new events associated with integrating IDS with region-based group key management.

We use the SPN model shown in Figure 5-1 (in Chapter 5) to describe the behaviors of a GCS instrumented with IDS to deal with insider attacks and region-based group key management to deal with outsider attacks in MANETs. The SPN model will be parameterized differently from that in Chapter 5 to account for the existence of the hierarchical group key management structure. Our goal is to identify optimal settings in terms of the optimal intrusion detection interval and the optimal regional area size to maximize *MTTSF* while satisfying imposed performance requirements in terms of $\hat{C}_{total}$.

8.2.2 Calculations of Metrics and Model Parameterization

MTTSF is obtained as by Equation 5-7 based on the concept of *mean time to absorption* in the SPN model. Specifically, we use a reward assignment such that a reward of 1 is assigned to all states except absorbing states which is modeled based on the two security failure conditions (i.e., if either Condition C1 or Condition C2 is met, the system fails). $\boldsymbol{\hat{C}_{total}}$ is calculated by the probability-weighted average of $\hat{C}_{total,i}$ representing the communication cost incurred per time unit (*s*) in state *i*. Specifically, $\hat{C}_{total}$ is calculated by accumulating $\hat{C}_{total,i}$ *(t)* over *MTTSF* divided by *MTTSF*, i.e.,

$$\hat{C}_{total} = \frac{\int_0^{MTTSF} \hat{C}_{total,i}(t)dt}{MTTSF} \qquad (8\text{-}1)$$

where $\boldsymbol{\hat{C}_{total,i}}$ is calculated as:

$$\hat{C}_{total,i} = \hat{C}_{GC,i} + \hat{C}_{status,i} + \hat{C}_{rekey,i} + \hat{C}_{IDS,i} + \hat{C}_{beacon,i} + \hat{C}_{mp,i} + \hat{C}_{mobility,i} \qquad (8\text{-}2)$$

where $\hat{C}_{GC,i}$, $\hat{C}_{status,i}$, $\hat{C}_{rekey,i}$, $\hat{C}_{IDS,i}$, $\hat{C}_{beacon,i}$, $\hat{C}_{mp,i}$, and $\hat{C}_{mobility,i}$ are the cost components for group communication, status exchange, rekeying, intrusion detection, beacon, group partition/merge, and mobility events, respectively, given that the number of groups in the system is *i*. Of these,

$\hat{C}_{GC,i}$, $\hat{C}_{beacon,i}$, $\hat{C}_{mp,i}$, and $\hat{C}_{mobility,i}$ follow the same way of calculations as in Chapter 7. Below we explain how to calculate $\hat{C}_{status,i}$, $\hat{C}_{rekey,i}$, and $\hat{C}_{IDS,i}$.

- $\boldsymbol{\hat{C}_{status,i}}$**:** this cost is for group node *status exchange for IDS*, calculated as:

$$\hat{C}_{status,i} = \frac{(N \times b_s) \times \left[N_{region,i} \times H_{region} + H_{leader,i}\right]}{T_{status}} \tag{8-3}$$

where T_{status} is the periodic time interval for disseminating a status exchange message, N is the number of current group members, and b_s is the message size (*bits*) of the status exchange information.

- $\boldsymbol{\hat{C}_{rekey,i}}$**:** this cost is for *group key rekeying due to group join/leave events and forced evictions* to evict detected compromised nodes, calculated as:

$$\hat{C}_{rekey,i} = \hat{C}_{join/leave,i} + \hat{C}_{eviction,i} \tag{8-4}$$

where $C_{join/leave,\,i}$ is the cost introduced by leave and join operations per time unit and $\hat{C}_{eviction,\,i}$ is the cost introduced by forced evictions per time unit. $\hat{C}_{join/leave,\,i}$ is as calculated in Chapter 7 and is not repeated here. $\hat{C}_{eviction,\,i}$ is calculated by:

$$\hat{C}_{eviction,i} = [rate(T\_IDS) + rate(T\_FA)] \times \hat{C}_{leave,i} \tag{8-5}$$

where *rate(T_IDS)* gives the intrusion detection rate and *rate(T_FA)* gives the false alarm rate which detects trusted nodes as compromised nodes, both can be obtained easily from the SPN model. $\hat{C}_{leave,i}$ is the cost per leave operation when there are *i* groups in the system, as calculated in Chapter 7.

- $\boldsymbol{\hat{C}_{IDS,i}}$**:** this is the *communication cost due to IDS*. For voting-based IDS, this cost is computed as:

$$\hat{C}_{IDS,i} = D(m_d) \times \left(1 - P_{fn}\right) \times N \times [b_{m-list} + m \times b_v] \times \left[H^{uni}_{leader,i} + H_{region} \times N_{region,i}\right] \tag{8-6}$$

where $D(m_d)$ is the detection rate based on the linear periodic detection function, P_{fn} is the probability of false negatives, N is the number of current members in a group, m is the number of vote-participants against a target node, $b_{m\text{-}list}$ is the message size (*bits*) of the list containing m vote participants, b_v is the message size (*bits*) of a vote, and $H^{uni}_{leader,i}$ is the number of hops between two leaders ($=r/R\sqrt{i}$).

The same parameterization principles explained in Chapter 5 (Section 5.2.3) are used to parameterize the group merging/partitioning rates ($N_{nm,i}$ and $N_{np,i}$) in Equation 5-7, number of active members (N), rekeying communication cost (T_{cm}) in Equation 5-3, linear attacker function ($A(m_c)$) in Equation 5-4, linear detection function ($D(m_d)$) in Equation 5-5, and false positive/negative probabilities (P_{fn} and P_{fp}) in Equation 5-6. When applying Equation 5-6 to obtain P_{fn} and P_{fp} for region-voting-based IDS, we use $N_{good} = mark(T_m) / N_{region,\ i}$ and $N_{bad} = mark(UC_m) / N_{region\ i}$. Here we consider the linear attacker function and the linear detection function without loss of generality. For selecting *m* vote participants, we restrict the selection of nodes that are in the same region of the target node for region-voting-based IDS, and nodes in the whole operational area for group-voting-based IDS. Without loss of generality, we also exemplify our region-based key agreement protocol with GDH, thus applying Equations 7-7 and 7-8 for the intra and inter-regional communication costs, respectively.

8.3 Numerical Results

In this section, we present numerical data for *MTTSF* and $\hat{C}_{total}$ obtained when given a set of parameter values characterizing the operational and environmental conditions and show that there exist optimal design settings in terms of the intrusion detection interval and the regional area size under which *MTTSF* is maximized while $\hat{C}_{total}$ is minimized for GCSs in MANET environments.

We also demonstrate the effectiveness of the integration design by performing a comparative analysis against a baseline model in which no region-based group key management is used, i.e., *n*=0, such that group-voting-based IDS must be used for intrusion detection.

Table 8-1: Main Design Parameters and Their Default Values.

Parameter	Value	Parameter	Value	Parameter	Value
λ	1/(60*60)	σ	1/(60*60*32)	T_{RB}	5 *s*
μ	1/(60*60*4)	BW	1Mbps	T_{LB}	2 *s*
T_{IDS}	5 – 1200 *s*	N_{init}	150 nodes	m	3
T_{status}	300 *s*	$D(m_d)/A(m_c)$	Linear	R	200 *m*
λ_c	1/(60*60*24)	r	500 *m*	b_{vote}	100 *bits*
λ_q	1/30	b_{GDH}	64 *bits*	$b_{m\text{-}list}$	100 *bits*
$p1$	2 %	b_{GC}	800 *bits*	M_{alive}	32 *bits*
$p2$	2 %	b_s	400 *bits*	U_{view}	500 *bits*

We vary values of key design parameters to analyze their effects on system performance. Table 8-1 summarizes default parameter values used in this case study. In particular, we use *p1* = *p2* = 1% since in general less than 1% of false positive or false negative rate is deemed acceptance.

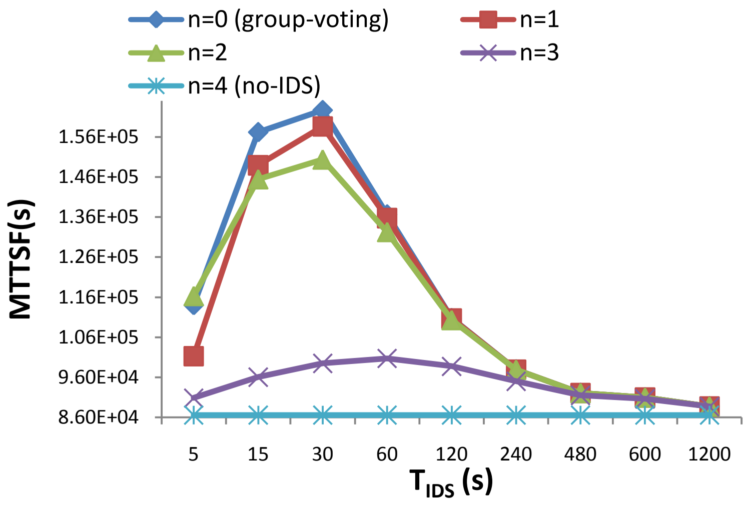


Figure 8- 1: *MTTSF* versus T_{IDS} with *p1* = *p2* = 0.005.

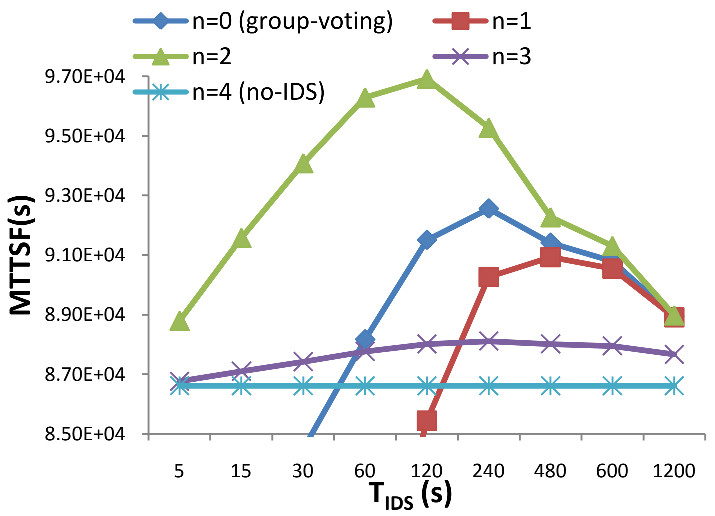


Figure 8- 2: *MTTSF* versus T_{IDS} with *p1* = *p2* = 0.02.

Figures 8-1 and 8-2 analyze the optimal settings in terms of T_{IDS} and *n* (representing the regional size) under which *MTTSF* is maximized. Figure 8-1 is for the case of low *p1* and *p2* values while Figure 8-2 is for the case of high *p1* and *p2* values. We first note that *MTTSF* obtained would be the same for group-voting-based IDS regardless of the regional size. Further, *MTTSF* obtained by group-voting-based IDS is exactly the same as that obtained by region-voting-based IDS when *n*=0 at which there is only a single region. In Figures 8-1 and 8-2, we label group-voting-based IDS corresponds to the special case of region-voting-based IDS in which there is only one region, i.e., n=0.

From Figures 8-1 and 8-2, we observe that there exists an optimal T_{IDS} that maximizes *MTTSF*. In general, as T_{IDS} increases, *MTTSF* increases until its optimal T_{IDS} is reached, and then *MTTSF* decreases after the optimal T_{IDS}. The reason of decreasing *MTTSF* after reaching the optimal point is that the false positive probability (P_{fp}) increases as T_{IDS} decreases, resulting in more nodes being falsely identified as compromised and being evicted from the system. Next we observe that there exists an optimal regional area size (represented by n) which is largely dictated by $p1$ and $p2$ values. Note that P_{fp} is one aspect of false alarms generated by IDS, which increases if IDS is more frequently triggered. When $p1 = p2$ is sufficiently low (i.e., $p1 = p2 = 0.005$) as shown in Figure 8-1, the best *MTTSF* is found with $n = 0$ at $T_{IDS} = 30$ *s*. When $p1 = p2$ is sufficiently high (i.e., $p1 = p2 = 0.02$) as shown in Figure 8-2, the best *MTTSF* is identified with $n = 2$ at $T_{IDS} = 120$ *s*. The optimal *MTTSF* exists due to the tradeoff between the positive and adverse effects of performing IDS.

Specifically, when $p1=p2$ is sufficiently low, P_{fp} and P_{fn} are also sufficiently low due to the positive effect of high quality IDS, so the system benefits the best by using m vote-participants out of a large region, i.e., at $n = 0$. On the other hand, when $p1= p2$ is high, P_{fp} and P_{fn} are also sufficiently high due to the adverse effect of IDS, so the system benefits the best by using m vote-participants out of a moderately large region at $n = 2$ such that it still has a good chance of finding m vote-participants in the region without suffering too much from high P_{fp} and P_{fn}.

Lastly, we notice that as $p1=p2$ increases, the optimal T_{IDS} increases. This is again due to the tradeoff between the positive and adverse effects of IDS. When high $p1=p2$ is used, triggering IDS less frequently will generate less false alarms. On the other hand, when sufficiently low $p1=p2$ is used, triggering of IDS often will improve *MTTSF* with the high quality IDS without generating frequent false alarms. We notice that at $n = 4$ *MTTSF* is low and flat because of the very low probability of finding m vote-participants in a small region, resulting in little IDS being used in the system.

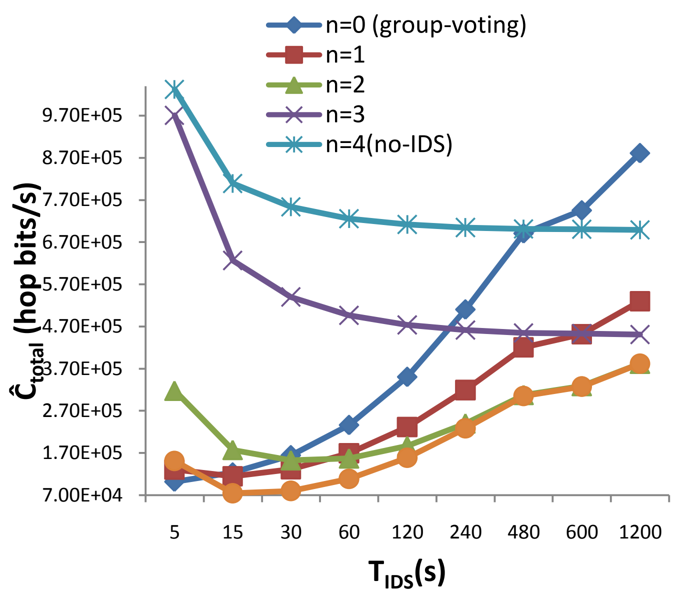


Figure 8- 3(a): $\hat{C}_{total}$ versus T_{IDS} with *p1 = p2 = 0.02*.

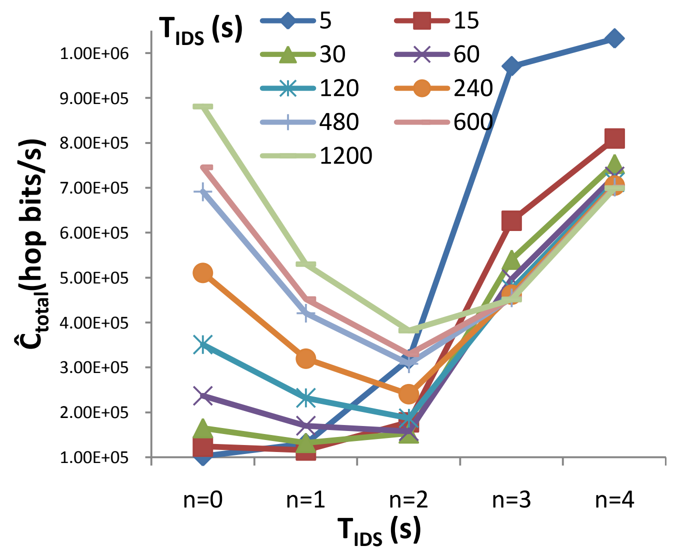


Figure 8-4-(b): $\hat{C}_{total}$ versus *n* with *p1 = p2 = 0.02*.

Next we analyze the optimal settings in terms of T_{IDS} and n (representing the regional size) under which $\hat{C}_{total}$ is minimized. Figure 8-3 (a) shows $\hat{C}_{total}$ versus T_{IDS} with varying n while Figure 8-3(b) shows $\hat{C}_{total}$ versus n with varying T_{IDS}. In Figure 8-3(a), we show $\hat{C}_{total}$ for group-voting-based IDS with n = 2 (optimal in terms of communication cost) for comparison purposes. Recall that for group-voting-based IDS, *MTTSF* is the same regardless of n because the probability of being able to find m vote-participants remains the same. However, $\hat{C}_{total}$ varies depending on n. We observe that for region-voting-based IDS, there is an optimal T_{IDS} under which $\hat{C}_{total}$ is maximized. Moreover, when T_{IDS} is sufficiently large, say $T_{IDS} > 30\ s$, the optimal n is 2. When $T_{IDS} = 15\ s$ or $30\ s$, the optimal n is at 1. The reason is that when T_{IDS} is sufficiently small, the node density tends to decrease rapidly because of frequent intrusion detection activities to evict compromised nodes. In this case, the system tends to favor a small number of regions (represented by $n = 1$) to reduce the inter-regional overhead, as in this case the inter-regional overhead will dominate the intra-regional overhead since the intra-regional overhead will be relatively small when the node density is low. Lastly when $n > 3$, $\hat{C}_{total}$ increases again because the inter-regional communications cost outweighs the intra-regional communication cost. We also observe that in Figure 8-4(a) group voting-based IDS at its optimal settings (T_{IDS} = 15, n=2) performs better than region-voting-based IDS at its optimal settings (T_{IDS} = 15, n=1). The reason is that group-voting-based IDS in its optimal setting could generate a longer *MTTSF* than region-voting-based IDS at its optimal setting. Consequently, group-voting-based IDS will generate a lower cost *per time unit* over its lifetime compared with region-voting-based IDS.

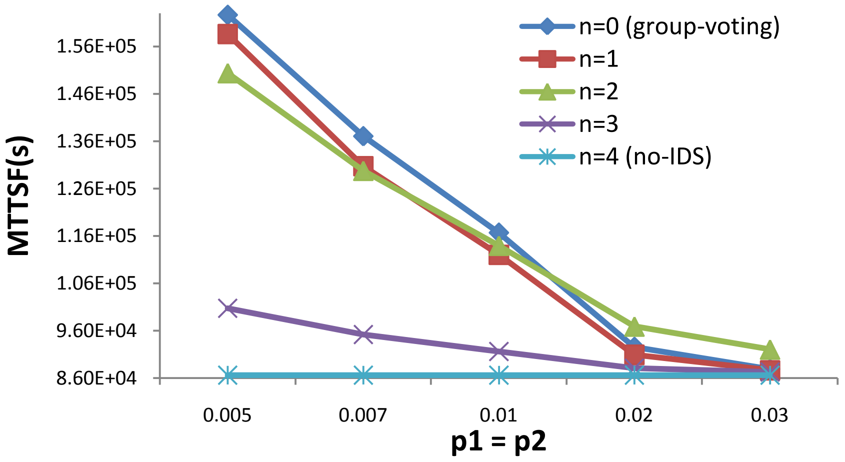


Figure 8- 4: *MTTSF* versus *p1* and *p2* with varying *n*.

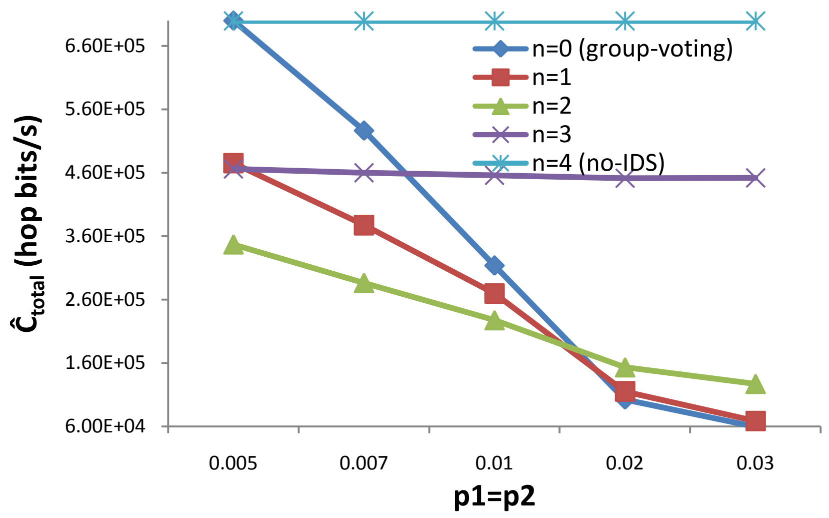


Figure 8- 5: $\hat{C}_{total}$ versus *p1* and *p2* with varying *n*.

Table 8- 2: Effect of λ_q and λ_c on an Optimal *n*, *MTTSF* and $\hat{C}_{total}$.

λ_q	*(n, MTTSF)*	*(n,* $\hat{C}_{total}$*)*
Once per 15 s	**(2, 92459)**	(0, 117913)
Once per 30 s	**(2, 96909)**	(0, 102351)
Once per 1 min	**(2, 103919)**	**(1, 92698)**
Once per 5 min	(0, 143663)	**(2, 47098)**
Once per 10 min	(0, 176235)	**(2, 37857)**
λ_c	*(n, MTTSF)*	*(n,* $\hat{C}_{total}$*)*
Once an hr	(0, 9676)	**(2, 401642)**
Once per 12 hrs	(0, 53392)	**(1, 168836)**
Once a day	(2, 96909)	**(0, 102351)**
Once per 2 days	(2, 458338)	**(0, 59462)**
Once per 4 days	(2, 20741200)	**(0, 37127)**

Next we analyze the effect of *p1* and *p2* on the optimal settings that maximize *MTTSF* and minimize $\hat{C}_{total}$. Figures 8-4 and 8-5 summarize the results. Here we fix T_{IDS} at its optimal value to isolate out of its effect. As shown in Figure 8-4, when *p1*= *p2* is sufficiently low (0.005 or 0.007 or 0.01), *MTTSF* is the best at *n* = 0. However, when *p1*= *p2* becomes higher, say *p1*= *p2* > 0.01, we observe the best *MTTSF* at *n* = 2. The results correlate well with the results presented earlier in Figures 8-1 and 8-2 so the same physical interpretation applies for the tradeoff between positive and negative effects of IDS applied. In Figure 8-5, we observe that when *p1*= *p2* is sufficiently low $\hat{C}_{total}$ is minimized at *n* = 2. However, when *p1*= *p2* becomes higher, say *p1*= *p2*

> 0.01, we observe that $\hat{C}_{total}$ is minimized at n = 1 or 0. Again this result correlates well with those presented in Figure 8-3. We note that there exists a tradeoff between security versus performance. While *MTTSF* is maximized at n = 2 when $p1 = p2$ is high (e.g., > 0.01), $\hat{C}_{total}$ is minimized at n = 2 only when $p1 = p2$ is low (e.g., ≤ 0.01). Consequently, the system designer should select the best settings under which the system requirements on *MTTSF* and $\hat{C}_{total}$ are best satisfied.

Finally we analyze the effect of the group communication rate (λ_q) and compromising rate (λ_c) on *MTTSF* or $\hat{C}_{total}$. Table 8-2 summarizes the optimal n value under which *MTTSF* is maximized and/or $\hat{C}_{total}$ is minimized. We fix all other parameter values at their default. We observe that as λ_q increases, the optimal n value increases, while the resulting *MTTSF* decreases. The reason for a low *MTTSF* when λ_q is high is due to high security failure in C1. We also observe that as λ_q increases, $\hat{C}_{total}$ increases because of the increased group communication cost.

For the effect of λ_c, we observe that as λ_c increases, the optimal n value decreases while the resulting *MTTSF* decreases. The reason that as λ_c increases, the optimal n value for maximizing *MTTSF* decreases is that with a high λ_c there are more compromised nodes in the system and the system will benefit more from using a smaller n to increase the probability of being able to find m vote-participants in order to more effectively evict compromised nodes. We also observe that when λ_c increases the optimal n in minimizing $\hat{C}_{total}$ increases while the resulting $\hat{C}_{total}$ increases slightly. The reason that $\hat{C}_{total}$ increases slightly as λ_c increases is that when there are more compromised nodes in a group, IDS is triggered more frequently, thus increasing the overall cost. The reason that the optimal n value increases when λ_c increases is that the inter-regional communication overhead in $\hat{C}_{IDS,i}$ (disseminating a message from a member to a leader) is small compared with the intra-regional communication overhead (disseminating a message from each leader to its regional members), so the system favors a large n to reduce the intra-regional overhead. Here again we see that a tradeoff exists between security versus performance. The system designer should select the best n such that both the security requirement (in terms of *MTTSF*) and performance requirement (in terms of $\hat{C}_{total}$) can be best satisfied.

8.4 Summary

In this chapter, we proposed and analyzed the design of integrating IDS with region-based group key management protocols for QoS-aware GCSs in multi-hop MANETs. In particular, we analyzed the effect of region-voting-based IDS versus group-voting-based IDS on security and performance properties of GCSs. Our results showed that there exist optimal settings in terms of the optimal regional area size and the IDS intrusion detection interval to maximize *MTTSF* while minimizing the total communication traffic. We demonstrated that under the optimal conditions indentified, the system with the integration design can provide a higher mean time to security failure while minimizing the total communication traffic, compared with a baseline system in which IDS is not integrated with region-based group key management.

Chapter 9 CONCLUSION

This dissertation research concerns the design and analysis of QoS-aware key management and intrusion detection protocols for secure mobile GCSs in wireless mobile networks. We have designed and evaluated three approaches for dealing with insider and outsider security attacks, namely, threshold-based periodic batch rekeying, intrusion detection, and region-based group key management for wireless networks with or without infrastructure support. The goal is to satisfy not only security requirements in terms of secrecy, confidentiality, authentication, availability, and data integrity, but also performance requirements in terms of latency, network traffic, response time, and reconfigurability for GCSs in wireless networks. These QoS-aware protocols are adaptive in nature with designs to allow the system to dynamically adjust operational settings, under which both the system's security and performance requirements can be best satisfied, leveraging the inherent tradeoff between performance and security goals.

Our proposed voting-based IDS protocol is independent of IDS mechanisms used. Existing host-IDS mechanisms in practice, such as misuse-based IDS or anomaly-based IDS, can be applied to realize host-based IDS preinstalled in each node. Voting-based IDS only requires host-based IDS be characterized by host-based false positive or false negative probabilities. While key management protocols developed in the dissertation research are mainly used to deal with outside attacks, they can also help mitigate insider attacks by means of rekeying to evict compromised nodes identified by IDS.

In Chapter 4, we investigated a class of QoS-aware threshold-based periodic batch rekeying protocols with the objective of reducing the communication overhead per join/leave operation (S) while satisfying delay (D) and secrecy (P_v) requirements for wireless GCSs where a centralized key server exists. We developed performance models to analyze these batch rekeying protocols, and compared their performance characteristics. We observed that an optimal batch rekey interval (T) exists for each of these protocols. We concluded that among the threshold-based periodic batch rekeying protocols proposed (*ULT*, *TAUDT*, and *JALDT*), *TAUDT* is able to produce the minimum S and the maximum T, which makes it the most efficient protocol among all three. As P_t increases, we observed a decreasing minimum S and an increasing T. As μ increases, we observed an increasing minimum S, and a decreasing optimal T. These results can be used by system designers to determine the optimal *T* value that would minimize *S* under *TAUDT*.

In Chapter 5, we proposed and analyzed voting-based IDS against inside attackers for secure GCSs in MANETs. The intrusion detection interval (T_{IDS}) used by voting-based IDS can be adjusted based on the behavior of inside attackers. Our analysis revealed the intrinsic tradeoff between security (measured by the *MTTSF* metric) and performance (measured by the overall communication cost $\hat{C}_{total}$ metric). When given a GCS characterized by a set of parameter values, we showed that there exists an optimal detection interval (T_{IDS}) that maximizes *MTTSF* as well as satisfying the constraint on the communication traffic ($\hat{C}_{total}$). The existence of the optimal detection interval (T_{IDS}) for maximizing *MTTSF* is attributed to the tradeoff between the positive effect of IDS (i.e., identifying compromised nodes and evicting them properly to prolong system lifetime) versus the adverse effect of IDS (i.e., false negatives and false positives generated by IDS). The existence of the optimal detection interval (T_{IDS}) for minimizing $\hat{C}_{total}$ is attributed to the tradeoff between the IDS communication traffic versus the group communication traffic. We have also performed sensitivity analysis of *MTTSF* versus T_{IDS} with respect to a number of key parameters, including the system failure definition (SF1 versus SF2), the number of vote-participants selected for performing majority voting in voting-based IDS (*m*), the group communication rate (λ_{q}) and the base compromising rate (λ_{c}) in single-hop as well as multi-hop MANET environments. We have investigated three ways to perform IDS detection and how the system could adjust the IDS detection level in response to the attacker strength detected at runtime in order to maximize *MTTSF* and minimize $\hat{C}_{total}$ dynamically. We discovered that we could select the best detection function (*logarithmic*, *linear*, or *polynomial*) in response to the attacker strength to maximize *MTTSF* without experiencing much of the adverse effect of IDS. The results obtained in terms of *MTTSF* and $\hat{C}_{total}$ versus T_{IDS} allow the system designer to select the best intrusion detection interval (T_{IDS}) to maximize *MTTSF*, or minimize $\hat{C}_{total}$, depending on the security versus performance requirements, or to maximize *MTTSF* while satisfying the $\hat{C}_{total}$ performance requirements.

In Chapter 6, we investigated the design of integrating intrusion detection with batch rekeying to cope with both outsider and insider attacks for GCSs in MANETs, and analyzed the tradeoff between security and performance properties of the resulting GCS due to the use of these two protocols. We showed that there exist optimal settings in terms of rekeying thresholds (*k1* and *k2*) reflecting batch rekey intervals and intrusion detection intervals under which the system lifetime in terms of *MTTSF* can be maximized while satisfying imposed performance requirements in terms of the average service response time, $\bar{R}$. We also demonstrated the effectiveness of the approach by comparing the resulting system performance with that of a system with IDS integrated with individual rekeying.

In Chapter 7, we have proposed and analyzed a scalable and efficient region-based secure group key management protocol to support secure group communications in MANETs. The region-based group key management protocol proposed not only reduces network communication costs, but also preserves secrecy and survivability security properties. By using GDH as an exemplary key agreement protocol for group key generation and rekeying at the intra-regional and inter-regional (leader) levels, we discovered that there exists an optimal regional size that would minimize the overall network traffic, when given a set of parameter values characterizing the operational condition. The existence of the optimal regional size is due to a tradeoff between inter-regional versus intra-regional overheads.

In Chapter 8, we proposed and analyzed the design of integrating IDS with region-based group key management protocols for QoS-aware GCSs in multi-hop MANETs. In particular, we analyzed the effect of region-voting-based IDS versus group-voting-based IDS on security and performance properties of GCSs. Our results showed that there exist optimal settings in terms of the optimal regional area size and the IDS intrusion detection interval to maximize *MTTSF* while minimizing the total communication traffic. We demonstrated that under the optimal conditions indentified, the system with the integration design can provide a higher mean time to security failure while minimizing the total communication traffic, compared with a baseline system in which IDS is not integrated with region-based group key management.

9.1 Publication

The dissertation research has resulted in several publications [18-29]. The preliminary research work on threshold-based periodic batch rekeying was published in [21]. Following [21], the preliminary work has been extended and published in the *ACM/Springer Wireless Networks* journal [20]. In these two papers, we developed an analytical model to address the issue of how often batch rekeying should be performed. We proposed the design concept of threshold-based periodic batch rekeying and demonstrated that an optimal batch rekey interval exists for each protocol. We further compared these protocols to identify the best protocol that can minimize the communication cost of rekeying while satisfying application requirements, when given a set of parameter values characterizing the operational and environmental conditions of the system.

The preliminary work on QoS-aware intrusion detection for mobile GCSs was published in [19]. In this work, we assumed that there exists a centralized key server capable of key generation, distribution, and revocation. The main contribution of this paper is to show the tradeoff between security and performance properties of IDS in mobile group communicating environments. Particularly, we analyzed how often IDS should perform intrusion detection activities to effectively trade security off for performance, or vice versa, for the system to satisfy the application security and performance requirements. Using *MTTSF* for the system to reach a security failure state, and overall communication cost as performance metrics, we identified the optimal intrusion detection rate under which the *MTTSF* metric can be best traded off for the performance metric. Subsequently in [26], we removed the assumption of the existence of a central key server and extended the work to multi-hop MANET environments considering multiple groups existing in the system due to group merge or partition derived from node mobility or node failure. This paper also considered different strength levels of attackers and incorporated designs that allow the system to dynamically adjust the intrusion detection interval based on the accumulated number of compromised nodes to maximize *MTTSF* while satisfying the response time requirement for secure group communications. In [25, 27], we analyzed the effect of system failure definitions on the reliability and *MTTSF* of a GCS in MANET. We also examined the effect of optimal intrusion detection intervals on *MTTSF* as well as on the overall communication cost for both single-hop MANETs in which only a single group exists and multi-hop MANETs in which multiple groups exist in the system.

In [18], we proposed and analyzed a scalable and efficient region-based group key management protocol for secure group communications in MANETs. We showed that an optimal regional size exists to minimize the total network communication cost while satisfying the secrecy requirement by effectively trading inter-regional versus intra-regional group key management overheads. In [22], we extended [18] to consider the existence of multiple groups due to group partition/merge events caused by node mobility and dynamic membership change. The analytical model was refined to deal with this change. Further, the analytical results were validated through extensive simulation.

In [23, 29], we integrated QoS-aware intrusion detection protocols proposed in Chapter 5 with threshold-based periodic batch rekeying protocols proposed in Chapter 4. We investigated performance characteristics of GCSs in multi-hop MANETs that employ intrusion detection techniques for dealing with inside attackers tightly coupled with batch rekeying techniques for dealing with outside attackers. The objective was to identify optimal settings including the best intrusion detection interval and the best batch rekey interval under which the system lifetime (*MTTSF*) is maximized while satisfying performance requirements. We also compared our design with a baseline system using intrusion detection integrated with individual rekeying to demonstrate the effectiveness.

In [24, 28], we integrated QoS-aware intrusion detection protocols proposed in Chapter 5 with region-based group key management addressed in Chapter 7 in MANET environments. The integrated design leverages region-based distributed group key management for scalability to deal with outside attackers, while employing voting-based IDS techniques for QoS awareness to deal with inside attackers. In this work, we identified optimal settings in terms of the best regional area size and intrusion detection interval under which *MTTSF* is maximized while minimizing the total network traffic incurred due to IDS and group communication activities in MANETs.

9.2 Applicability

The design and optimization principles developed in the dissertation research, including threshold-based periodic batch rekeying, QoS-aware intrusion detection protocols, and region-based group key management have wide applicability in various types of wireless network environments with or without infrastructure support. The methodologies developed for exploring the intrinsic tradeoff between performance and security properties of mobile GCSs are also generically applicable. As an example, if we consider a network environment in which a centralized key server and network-based IDS are employed, we can simply replace P_{fp} and P_{fn} with *p1* and *p2* where *p1* and *p2* refer to false negative and false positive probabilities in network-based IDS. As another example, we can consider a centralized or decentralized rekeying algorithm instead of a distributed rekeying algorithm such as GDH, by simply redefining rekeying conditions based on the state information provided in the SPN model, e.g., based on the number of join/leave/eviction operations in a state. The SPN models developed in the dissertation research for evaluating performance versus security properties can be easily extended to reflect changes of environment conditions.

Our notion of group communication is basically "connectivity" oriented, that is, whether nodes are in the same group depends on if they are connected regardless of whether the physical environment is structure-constrained. The IDS techniques and batch rekeying techniques are developed for connectivity-oriented group communications, so they can be applied to physical environments with structural constraints. The region-based group key management protocol developed is based on the assumption of open environments, so the optimal condition derived, i.e., how many regions should exist to divide the environment, depends on the population distribution and free mobility of users in this environment. Thus, to apply to physical environments with structural constraints, the region-based group key management protocol developed in the dissertation will need to be modified to reflect physical constraints.

To apply the results obtained in the dissertation research, one can test a range of possible values of model parameters and build a table at static time listing the selection of the optimal intrusion detection interval, batch rekey interval, and regional area size for maximizing *MTTSF* and/or minimizing the overall communication cost or the service response time, given a set of parameter values characterizing the operational and environmental conditions. Then, at runtime the system can perform a table lookup operation to select the best design parameter values (i.e., intrusion detection interval, batch rekey interval, or regional area size) based on statistical information collected dynamically for characterizing the operational and environmental conditions of the GCS.

In cases simple table lookup does not work well because there is a mismatch of system assumptions versus reality or because not all possible combinations have been considered at design time, we could use the static table as the backbone and augment simple table lookup with alternate designs. One possible design is to apply extrapolation to deduce optimal settings for those combinations not considered in the table. Another approach is to apply autonomic control to learn and rebuild the table based on feedback of actual optimal settings detected at runtime. Over time the table may be refined to shape the data contained within. A future research area is to investigate the feasibility of these alternate designs.

9.3 Future Work

We outline future research areas which may be extended from the dissertation work, including, (a) investigating the feasibility of applying autonomic control principles [70, 102] to learn and rebuild the look-up table based on feedback of actual optimal settings detected at runtime; (b) performing empirical studies by injecting faults and attacks to the GCS to obtain security and performance measures out of the implemented systems for experimental validation of the analytical results obtained [78]; (c) exploring real-world military applications with mission-oriented GCSs in MANET environments to which the design principles and assessment methodologies developed in the paper can be applied [72]; (d) investigating the definitions of group formation, group failure and system failure and their effects on the system lifetime of GCSs [33, 62]; (e) investigating the effect of mobility model, e.g., group mobility model versus random waypoint model, as well as the group communication model, e.g., publish/subscribe multicast versus end-to-end, on *MTTSF* of GCSs [86]; (f) exploring trust-based IDS as opposed to voting-based IDS and discovering the best way to perform trust-based IDS for maximizing *MTTSF* while satisfying performance requirements for GCSs in MANET environments [37, 81, 87]; and (g) extending QoS-aware security protocols developed in the dissertation research to support non-repudiation services [11] for GCS applications based on single-source or multiple-source communication paradigms [15, 73, 109].

Bibliography

[1] Y. Amir, C. Nita-Rotaru, J. L. Schultz, J. Stanton, and G. Tsudik, "Secure Spread: An Integrated Architecture for Secure Group Communication," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 3, 2005, pp. 248-261.

[2] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the Performance of Group Key Agreement Protocols," *ACM Transactions on Information and System Security*, vol. 7, no. 3, 2004, pp. 457-488.

[3] Y. Amir, Y. Kim, C. Nita-Rotaru, J.L. Schultz, J. Stanton, and G. Tsudik, "Secure Group Communication Using Robust Contributory Key Agreement" *IEEE Transactions on Parallel and Distributed Systems,* vol. 15, no. 5, 2004, pp.468-480.

[4] S. Axelsson, "Intrusion Detection Systems: A Taxonomy and Survey," *Technical Report*, no. 99-15, Department of Computer Engineering, Chalmers University of Technology, Sweden, 20 March 2003.

[5] A. Balasubramanian, S. Mishra, and R. Sridhar, "Analysis of a Hybrid Key Management Solution for Ad hoc Networks," *Proceedings of 2005 IEEE Wireless Communications and Networking Conference*, vol. 4, March 2005, pp. 2082 - 2087.

[6] S. Banerjee and B. Bhattacharjee, "Scalable Secure Group Communication over IP Multicast", *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, Oct. 2002, pp. 1511-1527.

[7] S. Basagni, "Distributed Clustering for Ad Hoc Networks," *1999 International Symposium on Parallel Architectures, Algorithms and Networks*, IEEE Computer Society, Australia, 23-25 June 1999, pp. 310-315.

[8] M. Bechler, H.-J. Hof, D. Kraft, F. Pahlke, and L. Wolf, "A Cluster-Based Security Architecture for Ad Hoc Networks," *Proceedings of 23rd IEEE INFOCOM*, vol. 4, March 2004, pp. 2393- 2403.

[9] C. Becker and U. Wille, "Communication Complexity of Group Key Distribution," *Proceedings of 5th ACM Conference on Computer and Communications Security*, San Francisco, CA, 2-5 Nov. 1998, pp. 1-6.

[10] G. Bianchi, “Performance Analysis of the IEEE 802.11 Distributed Coordination Function,” *IEEE Journal on Selected Areas in Communications,* vol. 18, no. 3, March 2000, pp. 535-547.

[11] B. Briscoe and I. Fairman, “Nark: receiver-based multicast non-repudiation and key management,” *Proceedings of 1st ACM Conference on Electronic Commerce*, Denver, Colorado, Nov. 1999, pp. 22-30.

[12] P. Brutch and C. Ko, “Challenges in Intrusion Detection for Wireless Ad-hoc Networks,” *Proceedings of Symposium on Applications and the Internet Workshops*, Orlando, FL, 27-30 January 2003, pp. 368 – 373.

[13] M. Burmester and Y. Desmedt, “A Secure and Efficient Conference Key Distribution System,” *Advances in Cryptology: EROCRYPT94*, vol. 950, 1994, pp. 275-286.

[14] J.B.D. Cabrera, C. Gutierrez, and R. K. Mehra, “Infrastructures and Algorithms for Distributed Anomaly-based Intrusion Detection in Mobile Ad-hoc Networks,” *IEEE Military Communications Conference (MILCOM 2005)*, vol. 3, Oct. 2005, pp. 1831 – 1837.

[15] W. Caifen, G. Jianhua, D. Xinjun, Q. Jin, Z. Tieshan, and Y. Shiyong, “A Multi-party Non-repudiation Protocol with Semi-trusted Third Party,” *Proceedings of IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering*, vol. 1, 28-31 Oct. 2002, pp. 188-191.

[16] M.M. Carvalho and J.J. Garcia-Luna-Aceves, “Delay Analysis of IEEE 802.11 in Single-hop Networks,” *Proceedings of 11th IEEE International Conference on Network Protocols,* Atlanta, GA, Nov. 2003, pp. 146-155.

[17] H. Chan, V. D. Gligor, A. Perrig, and G. Muralidharan, “On the Distribution and Revocation of Cryptographic Keys in Sensor Networks,” *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no 3, July-Sep. 2005, pp. 233 – 247.

[18] I.R. Chen, J.H. Cho and D.C. Wang, “Performance Characteristics of Region-based Group Key Management in Mobile Ad Hoc Networks,” *Proceedings of 1st IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, Taichung, Taiwan, June 2006, pp. 411-419.

[19] J.H. Cho and I.R. Chen, “On Design Tradeoffs Between Security and Performance in Wireless Group Communicating Systems,” *Proceedings of IEEE 1st Workshop on Secure Network Protocols (NPSec),* Boston, Nov. 2005, pp. 13-18.

[20] J.H. Cho, I.R. Chen and M. Eltoweissy, "On Optimal Batch Rekey Interval for Secure Group Communications in Wireless Networks," *Wireless Networks (ACM/Springer),* vol. 14, no. 6, Dec. 2008, pp. 915-927.

[21] J.H. Cho, I.R. Chen and M. Eltoweissy, "Optimization of Batch Rekey Interval for Secure Group Communications in Wireless Networks," *Proceedings of 2005 IEEE International Conference on Wireless Networks, Communications, and Mobile Computing (Wirelesscom 2005),* Maui, Hawaii, USA, vol. 1, July 2005, pp.522-527.

[22] J.H. Cho, I.R. Chen, and D.C. Wang "Performance Optimization of Region-Based Group Key Management in Mobile Ad Hoc Networks," *Performance Evaluation (Elsevier),* vol. 65, no. 5, May 2008, pp. 319-344.

[23] J.H. Cho, I.R. Chen, and P.G. Feng, "Performance Analysis of Dynamic Group Communication Systems with Intrusion Detection Integrated with Batch Rekeying in Mobile Ad Hoc Networks," *Proceedings of 22nd International Conference on Advanced Information Networking and Applications-Workshop(AINAW2008),* GinoWan, Okinawa, Japan, 25-28 March 2008, pp. 644 - 649.

[24] J. H. Cho and I.R. Chen, "Effect of Intrusion Detection on Secure Group Communication in Hierarchically Structured Group Architectures," *Proceedings of 4th IEEE LCN Workshop on Network Security,* Montreal, Canada, Oct. 2008.

[25] J.H. Cho, I.R. Chen, and P.G. Feng, "Effect of Intrusion Detection on Failure Time of Mission-Oriented Mobile Group Systems in Mobile Ad Hoc Networks," *Proceedings of 14th IEEE Pacific Rim International Symposium on Dependable Computing,* Taipei, Taiwan, Dec. 2008.

[26] J.H. Cho and I.R. Chen, "Performance Analysis of Intrusion Detection Protocols in Mobile Group Communication Systems," submitted to *23rd IEEE International Parallel & Distributed Processing Symposium,* Oct. 2008.

[27] J.H. Cho, I.R. Chen, and P.G. Feng, "Effect of Intrusion Detection on Reliability of Mission-Oriented Mobile Group Systems in Mobile Ad Hoc Networks," submitted to *IEEE Transactions on Reliability,* Aug. 2008.

[28] J.H. Cho and I.R. Chen, "Adaptive Intrusion Detection for Region-based Hierarchical Group Key Management Protocols in Mobile Ad Hoc Networks," submitted to *Journal of Systems and Software (Elsevier),* Sept. 2008.

[29] J.H. Cho, I.R. Chen, and P.G. Feng, "Modeling and Analysis of Intrusion Detection Integrated with Batch Rekeying for Dynamic Group Communication Systems in Mobile Ad Hoc Networks," submitted to *ACM/Springer Wireless Networks,* Dec. 2007.

[30] G. Ciardo, R.M. Fricks, J.K. Muppala and K.S. Trivedi, *SPNP Reference Guide Version 4*, Department Electrical Engineering, Duke University, 1994.

[31] G. Ciardo, R.M. Fricks, J.K. Muppala and K.S. Trivedi, *SPNP Users Manual Version 6*, Department Electrical Engineering, Duke University, 1999.

[32] M. Dacier, Y. Deswarte, and M. Kaâniche, "Quantitative Assessment of Operational Security: Models and Tools," *Technical Report* 96493, Laboratory for Analysis and Architecture of Systems, May 1996.

[33] S. K. Das and S. Bhattacharya, "Prospects of Group-Based Communication in Mobile Ad Hoc Networks," *Distributed Computing, Lecture Notes in Computer Science,* Springer Berlin/Heidelberg, vol. 2571, 2002, pp. 174-183.

[34] H. Debar and A. Wespi, "Aggregation and Correlation of Intrusion-Detection Alerts," *Proceedings of 4th International Symposium Recent Advances in Intrusion Detection*, Davis, CA, Oct. 2001, pp. 85-103.

[35] L. Dondeti, S. Mukherjee, and A. Samal, "A Distributed Group Key Management Scheme for Security Many-to-Many Communication," *Technical Report,* PINTL-TR-207-99, Department of Computer Science, University of Maryland, 1999.

[36] L. Doneti, S. Mukherjee, and A. Samal, "Scalable Secure One-to-Many Group Communication using Dual Encryption," *Computer Communications*, vol. 23, no. 17, Nov. 2000, pp. 1681-1701.

[37] C. Duma, M. Karresand, N. Shahmehri, and G. Caronni, "A Trust-Aware, P2P-Based Overlay for Intrusion Detection," *Proceedings of 17th International Conference on Database and Expert Systems Applications (DEXA),* Washington D.C., 4-8 Sep. 2006, pp. 692-697.

[38] C. Duma, N. Shahmehri, and P. Lambrix, "A Hybrid Key Tree Scheme for Multicast to Balance Security and Efficiency Requirements," *Proceedings of 12th International Workshop Enabling Technologies: Infrastructure for Collaborative Enterprises*, Linz, Austria, 9-11 June 2003, pp. 208-213.

[39] M. Eltoweissy, M. Moharrum, and R. Mukkamala, "Dynamic Key Management in Sensor Networks," *IEEE Communications Magazine*, vol. 44, no. 4, 2006, pp. 122-130.

[40] F. C. Gärtner, "Byzantine Failures and Security: Arbitrary is Not (always) Random," Swiss Federal Institute of Technology (EPFL) School of Computer and Communication Sciences, *Technical Report* IC/2003/20, 2003.

[41] A. Ghosh, and F. Anjum, "Wireless Network Security II: Last Hop Topology Sensitive Multicasting Key Management," *The 1st ACM International Workshop on Quality of Service and Security in Wireless and Mobile Networks*, Montreal, Quebec, Canada, Oct. 2005, pp. 63-70.

[42] T. Hardjono, B. Cain, and I. Monga, "Intra-Domain Group Key Management Protocol," *Internet Draft*, Feb. 1998.

[43] H. Harney and C. Muckenhirn, "Group Key Management Protocol (GKMP) Specification," Report No. RFC 2093, 1997.

[44] H. Harney and C. Muckenhirn, "Group Key Management Protocol (GKMP) Architecture," Report No. RFC 2094, 1997.

[45] Y. Huang and W. Lee, "A Cooperative Intrusion Detection System for Ad Hoc Networks," *Proceedings of 1st ACM Workshop on Security of Ad-hoc and Sensor Networks,* Fairfax, VA, Oct. 2003, pp. 135-147.

[46] I. Ingemarsson, D.T. Tang, and C.K. Wong, "A Conference Key Distribution System," *IEEE Transactions on Information Theory*, vol. 28, no. 5, Sep. 1982, pp. 714-720.

[47] K. Inkinen, "New Secure Routing in Ad Hoc Networks: Study and Evaluation of Proposed Schemes," *Seminar on Internetworking*, Sjökulla, Finland, spring 2004.

[48] E. Jonsson and T. Olovsson, "A Quantitative Model of the Security Intrusion Process Based on Attacker Behavior," *IEEE Transactions on Software Engineering*, vol. 23, no. 4, April 1997, pp. 235-245.

[49] E. Jung, A.X. Liu, and M.Gouda, "Key Bundles and Parcels: Secure Communication in Many Groups," *Computer Networks*, vol. 50, no. 11, Aug. 2006, pp. 1781-1798.

[50] O. Kachirski and R. Guha, "Intrusion Detection Using Mobile Agents in Wireless Ad Hoc Networks," *Proceedings of IEEE Workshop on Knowledge Media Networking*, July 2002, pp. 153-158.

[51] T. Karygiannis and L. Owens, "Wireless Network Security: 802.11, Bluetooth and Handheld Devices," National Institute of Standards and Technology (NIST), Special Publication 800-48, 2002.

[52] P. Kazienko and P. Dorosz, Intrusion Detection Systems (IDS) Part I: Network Intrusions, Attack Symptoms, IDS Tasks, and IDS Architecture, 2004, http://www.windowsecurity.com/articles/intrusion_detection/.

[53] P. Kazienko and P. Dorosz, Intrusion Detection Systems (IDS) Part 2: Classification, Methods, Techniques, 2004, http://www.windowsecurity.com/articles/intrusion_detection/.

[54] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based Group Key Agreement," *ACM Transactions on Information and System Security,* vol. 7, no. 1, 2004, pp. 60-96.

[55] Y. Kim, A. Perrig and G. Tsudik, "Communication-Efficient Group Key Agreement," *Proceedings of IFIP TC11 16th Annual Working Conference on Information Security*," vol. 193, June 2001, pp. 229-244.

[56] L. Lazos and R. Poovendran, "Location-Aware Secure Wireless Multicast in Ad hoc Networks under Heterogeneous Pathloss," UWEETR-2003-0012, *UWEE Technical Report Series*, 2003.

[57] L. Lazos and R. Poovendran, "Energy-Aware Secure Multicast Communication in Ad-hoc Networks Using Geographic Location Information," *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing*, vol. 4, April 2003, pp. 201-204.

[58] P.C. Lee, C.S. Lui, and K.Y. Yau, "Distributed Collaborative Key Agreement Protocols for Dynamic Peer Groups," *Proceedings of 10th IEEE International Conference on Network Protocols*, Paris, France, 4-7 Nov. 2002, pp. 322 – 333.

[59] D.J. Leversage and E. James, "Estimating a System's Mean Time-to-Compromise," *IEEE Security and Privacy*, vol. 6, no.1, Jan.-Feb. 2008, pp. 52-60.

[60] X. Li, Y.R. Yang, M.G. Gouda, and S.S. Lam, "Batch Rekeying for Secure Group Communications," *Proceedings of 10th International Conference on World Wide Web*, Hong Kong, Hong Kong, May 2001, pp.525-534.

[61] J. H. Li, R. Levy, M. Yu and B. Bhattacharjee, "A Scalable Key Management and Clustering Scheme for Ad hoc Networks," *Proceedings of 1st ACM International Conference on Scalable Information Systems*, vol. 152, no. 28, Hong Kong, Hong Kong, May 2006, pp. 1-10.

[62] J. Liu, D. Sacchetti, F. Salhan, and V. Issarny, "Group Management for Mobile Ad Hoc Networks: Design, Implementation, and Experiment," *Proceedings of 6th International Conference on Mobile Data Management,* Ayia Napa, Cyprus, 9-13 May 2005, pp. 192-199.

[63] B.B. Madan, K.G. Popstojanova, K. Vaidyanathan, and K.S. Trivedi, "A Method for Modeling and Quantifying the Security Attributes of Intrusion Tolerant Systems," *Performance Evaluation,* vol. 56, no. 1-4, 2004, pp. 167-186.

[64] B. Madan, K. G. Popstojanova, K. Vaidyanathan, and K.S. Trivedi, "Modeling and Quantification of Security Attributes of Software Systems," *International Conference Dependable Systems and Networks,* Washington D.C., June 2002, pp. 505-514.

[65] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," *Proceedings of 6th Annual ACM/IEEE Mobile Computing and Networking*, Boston, MA, Aug. 2000, pp.255-265.

[66] D. A. McGrew and A. T. Sherman, "Key Establishment in Large Dynamic Groups using One-way Function Trees," *Technical Report,* no. 0755, TIS Labs at Network Associates, Inc. 1998.

[67] A. Mishra, K. Nadkarni, and A. Patcha, "Intrusion Detection in Wireless Ad-hoc Networks," *IEEE Wireless Communications,* vol. 11, no. 1, Feb. 2004, pp.48-60.

[68] S. Mittra, "Iolus: A Framework for Scalable Secure Multicasting," *Proceedings of ACM SIGCOMM,* vol. 27, no. 4, Cannes, France, Oct. 1997, pp.277-288.

[69] M. Moharrum, R. Mukkamala, and M. Eltoweissy, "Efficient Secure Multicast with Well-Populated Multicast Key Trees," *Proceedings of 10th International Conference on Parallel and Distributed Systems,* 7-9 July 2004, pp.215-222.

[70] R. Mortier and E. Kiciman, "Autonomic Network Management: Some Pragmatic Considerations," *Proceedings of 2006 SIGCOMM Workshop on Internet Network Management (INM'06),* Pisa, Italy, 11-15 Sep. 2006, pp. 89-93.

[71] D.M. Nicol, W.H. Sanders, and K.S. Trivedi, "Model-Based Evaluation: From Dependability to Security," *IEEE Transactions on Dependability and Secure Computing*, vol. 1, no.1, Jan.-March 2004, pp. 48-65.

[72] E. Onem, H. B. Yilmaz, F. Alagöz, and T. Tugcu, "On Communication Protocols for Tactical Navigation Assistance," *Proceedings of 1st International Conference on Mobile*

Wireless Middleware, Operating Systems, and Applications, Innsbruck, Austria, vol. 278, no. 39, 12-15 Feb. 2008.

[73] J. A. Onieva, J. Zhou, and J. Lopez, "Non-repudiation Protocols for Multiple Entities, *Computer Communications*, vo. 27, no. 16, 15 Oct. 2004, pp. 1608-1616.

[74] A. Perrig and J.D. Tygar, *Secure Broadcast Communication in Wired and Wireless Networks*, Kluwer Academic Publishers, 2002.

[75] R. D. Pietro, L. V. Mancini, and S. Jajodia, "Security and Middleware Services: Efficient and Secure Keys Management for Wireless Mobile Communications," *Proceedings of 2nd ACM International Workshop on Principles of Mobile Computing,* Toulouse, France, Oct. 2002, pp. 66-73.

[76] R. D. Pietro, L. V. Mancini, Y. W. Law, S. Etalle, and P. Havinga. "LKHW: A Directed Diffusion-based Secure Multicast Scheme for Wireless Sensor Networks," *Proceedings of 1st International Workshop on Wireless Security and Privacy*, Oct. 2003, pp. 397-406.

[77] K. G. Popstojanova, F. Wang, R. Wang, F. Gong, K. Vaidyanathan, K.S. Trivedi, and B. Muthusamy, "Characterizing Intrusion Tolerant Systems Using a State Transition Model," *Proceedings of DARPA Information Survivability Conference and Exposition,* vol. 2, June 2001, pp. 211-221.

[78] H. V. Ramasamy and P. Pandey, J. Lyons, M. Cukier, and W. H. Sanders, "Quantifying the cost of Providing Intrusion Tolerance in Group Communication Systems," *Proceedings of International Conference on Dependable Systems and Networks*, 23-26 June 2002, pp. 229-238.

[79] S. Rafaeli and D. Hutchison, "A Survey of Key Management for Secure Group Communication," *ACM Computing Surveys (CSUR),* vol. 35, no. 3, Sep. 2003, pp. 309-329.

[80] S. Rafaeli and D. Hutchison, "HYDRA: A Decentralized Group Key Management," *Proceedings of 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2002),* Pittsburg, PA, 10-12 June 2002, pp. 62-67.

[81] M. Rehak, M. Pechoucek, P. Celeda. J. Novotny, and P. Minarik, "CAMNEP: Agent-based Network Intrusion Detection System," *Proceedings of 7th International Joint Conference on Autonomous Agents and Multiagent Systems,* Estoril, Portugal, 12-16 May 1008, pp.133-136.

[82] K. H. Rhee, Y. H. Park, and T. Gene, “An Architecture for Key Management in Hierarchical Mobile Ad-hoc Networks,” *Journal of Communications and Networks*, vol. 6, no. 2, 2004, pp.156-162.

[83] O. Rodeh, K. Birman, and D. Dolev, “Optimized Group Rekey for Group Communication Systems,” *Symposium Network and Distributed System Security*, San Diego, CA, Feb. 2000.

[84] R.A. Sahner, K.S. Trivedi and A. Puliafito, *Performance and Reliability Analysis of Computer Systems*, Kluwer Academic Publishers, 1996.

[85] R. Sandhu, Lecture Note: Internet Security Protocols, http://www.list.gmu.edu/infs766/infs766sp04ng/l1-2-6.pdf

[86] P. Santi, “Topology Control in Wireless Ad Hoc and Sensor Networks,” *ACM Computing Survey (CSUR)*, vol. 37, no. 2, June 2005, pp. 164-194.

[87] P. Sen, N. Chaki, and R. Chaki, “HIDS: Honesty-Rate Based Collaborative Intrusion Detection System for Mobile Ad Hoc Networks,” *Proceedings of 7th International Conference on Computer Information Systems and Industrial Management Applications*, 26-28 June 2008, pp. 121-126.

[88] A. Seshadri, M. Luk, A. Perrig, L. Van Doorn, P. Khosla, “SCUBA: Secure Code Update by Attestation in Sensor Networks,” *Proceedings of 5th ACM Workshop on Wireless Security*, Los Angeles, CA, Sep. 2006, pp.85-94.

[89] S. Setia, S. Koussih, S. Jajodia, and E. Harder, “Kronos: A Scalable Group Re-keying Approach for Secure Multicast”, *IEEE Symposium on Security and Privacy*, Oakland, CA, 14-17 May 2000, PP.215-228.

[90] S. Singh, M. Cukier, and W.H. Sanders, “Probabilistic Validation of an Intrusion-Tolerant Replication System,” *Proceedings of International Conference Dependable Systems and Networks*, 22-25 June 2003, pp. 615-624.

[91] M. Steiner, G.Tsudik, and M. Waidner, “Key Agreement in Dynamic Peer Groups,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 8, 2000, pp. 769-980.

[92] M. Steiner, G. Tsudik, and M. Waidner, “Diffie-Hellman Key Distribution Extended to Group Communication,” *Proceedings of 3rd ACM Conference on Computer and Communications Security*, New Delhi, India, Jan. 1996, pp. 31-37.

[93] M. Steiner, G. Tsudik, and M. Waidner, "CLIQUES: A New Approach to Group Key Agreement," *Proceedings of 18th International Conference on Distributed Computing Systems,* Washington D.C., May 1998, pp. 380-387.

[94] D. Sterne, P. Balasubramanyam, D. Carman, B. Wilson, R. Talpade, C. Ko, R. Balupari, C. Y. Tseng, and T. Bowen, "A General Cooperative Intrusion Detection Architecture for MANETs," *Proceedings of 3rd IEEE International Workshop on Information Assurance*, College Park, MD, 23-24 March 2005, pp. 57-70.

[95] F. Stevens, T. Courtney, S. Singh, A. Agbaria, J.F. Meyer, W.H. Sanders, and P. Pal, "Model-Based Validation of an Intrusion-Tolerant Information System," *Proceedings of 23rd Symposium Reliable Distributed Systems,* Nürnberg, Germany, 18-20 Oct. 2004, pp.184-194.

[96] D. Subhadrabandhu, S. Sarkar, and F. Anjum, "Efficacy of Misuse Detection in Ad-hoc Networks," *Proceedings of 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks,*4-10 Oct. 2004, pp. 97-107.

[97] D. Subhadrabandhu, S. Sarkar, and F. Anjum, "A Framework for Misuse Detection in Ad Hoc Networks- part I," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, Feb. 2006, pp. 274-289.

[98] D. Subhadrabandhu, S. Sarkar, and F. Anjum, "A Framework for Misuse Detection in Ad Hoc Networks- Part II," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, Feb. 2006, pp. 290-304.

[99] B. Sun, K. Wu, and U. W. Pooch, "Alert Aggregation in Mobile Ad Hoc Networks," *Proceedings of 2003 ACM Workshop on Wireless Security*, San Diego, CA, Sep. 2003, pp. 69-78.

[100] B. Sun, K. Wu, and U.W. Pooch, "Routing Anomaly Detection in Mobile Ad-hoc Networks," *Proceedings of IEEE 12th International Conference on Computer Communications and Networks (ICCCN2003)*, Oct. 2003, pp. 25-31.

[101] B. Sun, L. Osborne, Y. Xiao, S. and Guizani, "Intrusion Detection Techniques in Mobile Ad hoc and Wireless Sensor Networks," *IEEE Wireless Communications*, vol. 14, no. 5, Oct. 2007, pp. 56-63.

[102] A. Tizgadam and A. Leon-Garcia, "AORTA: Autonomic Network Control and Management System," *Proceedings of IEEE Conference on Computer Communications Workshops, INFOCOM,* Phoenix, AZ, 13-18 April 2008, pp. 1-4.

[103] Y. Wang, J. Li, L. Tie, and Q. Li, "An Efficient Key Management for Large Dynamic Groups," *Proceedings of 2nd Annual Conference on Communication Networks and Service Research*, May 2004, pp. 131-136.

[104] Y. Wang, X. Li, and O. Frieder, "Efficient Hybrid Key Agreement Protocol for Wireless Ad Hoc Networks," *Proceedings of 11th International Conference on Computer Communications and Networks (ICCCN'02)*, Miami, FL, 14-16 Oct. 2002, pp. 404–409.

[105] D. Wang, D.W. Bharat, B. Madan, and K.S. Trivedi, "Security Analysis of SITAR Intrusion Tolerance System," *Proceedings of 2003 ACM Workshop on Survivable and Self-regenerative Systems,* Fairfax, VA, Oct. 2003, pp. 23-32.

[106] J.W. Wilson and I.R. Chen, "Performance Characteristics of Location-based Group Membership and Data Consistency Algorithms in Mobile Ad hoc Networks," *International Journal of Wireless and Mobile Computing,* vol. 1, no. 8, 2005.

[107] C.K. Wong, M. Gouda, and S.S. Lam, "Secure Group Communications Using Key Graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, Feb. 2000, pp. 16-30.

[108] S. Xu, Z. Yang, Y. Tan, W. Liu, and S. Sesay, "An Efficient Batch Rekeying Scheme based on One-way Function Tree," *IEEE International Symposium on Communications and Information Technology (ISCIT2005)*, vol. 1, 2005, pp. 490-493.

[109] W. Xueming and L. Xiang, "Game-Based Analysis of Multi-Party Non-Repudiation Protocols, *International Conference on Computational Intelligence and Security*, Harbin, China, 15-19 Dec. 2007, pp. 642-646.

[110] Y.R. Yang, X. Li, X. Zhang, and S.S. Lam, "Reliable Group Rekeying: A Performance Analysis," *Proceedings of ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM 2001*, San Diego, CA, Aug. 2001, pp. 27-38.

[111] M. Younis, K Ghumman, and M. Eltoweissy, "Location-Aware Combinatorial Key Management Scheme for Clustered Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 8, Aug. 2006, pp.865-882.

[112] Y. Zhang, W. Lee, and Y.A. Huang, "Intrusion Detection Techniques for Mobile Wireless Networks," *Wireless Networks,* vol. 9, 2003, pp. 545-556.

[113] Y. Zhang and W. Lee, "Intrusion Detection in Wireless Ad Hoc Networks," *Proceedings of 6th International Conference Mobile Computing and Networks*, Aug. 2000, pp. 275–83.

[114] C. Zhang, B. DeCleene, J. Kurose, and D. Towsley, "Comparison of Inter-Area Rekeying Algorithms for Secure Wireless Group Communications," *Performance Evaluation,* vol. 49, no. 1-4, 2002, pp. 1-20.

[115] L. Zhou and Z. Haas, "Securing Ad Hoc Networks," *IEEE Network Magazine, Special Issues on Networking Security*, vol. 13, no. 6, Nov./Dec. 1999, pp. 24-30.

[116] National Security Information Assurance Solutions Technical Directors, "Information Assurance Technical Framework (IATF)," Release 3.1, Sep. 2002, http://www.iatf.net/framework_docs/version-3_1/index.cfm.

[117] "Standard Error" definition from *Wikipedia* http://en.wikipedia.org/wiki/Standard_error_(statistics).

Appendix A: Summary of Acronyms

ACK	Acknowledgement
BD	Burmester and Desmedt
CH	Cluster Head
CKA	Contributory Key Agreement
CRN	Common Random Number
CTS	Clear-To-Send
CW	Contention Window
DCF	Distributed Coordination Function
DEP	Dual Encryption Protocol
DH	Diffie-Hellman
DIFS	Distributed Inter Frame Space
DLKH	Distributed Logical Key Hierarchy
DOFT	Distributed One-way Function Tree
DoS	Denial-of-Service
DSSS	Direct Sequence Spread Spectrum
GCS	Group Communication System
GDH	Group Diffie-Hellman
GPS	Global Positioning System
HKT	Hybrid Key Tree
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
ING	INGemarsson
IETF	Internet Engineering Task Force
IGKMP	Intra-domain Group Key Management Protocol
JALDT	Join And Leave Double Threshold-based
LKH	Logical Key Hierarchy
LKHW	Logical Key Hierarchy for Wireless sensor network
MAC	Message Authentication Code
MAC	Medium Access Control

MANET	Mobile Ad Hoc Network
MPD	Mean Percentage Difference
MTTA	Mean Time To Absorption
MTTSF	Mean Time To Security Failure
OFT	One-way Function Tree
PKI	Public Key Infrastructure
RTS	Request-To-Send
RWM	Random Waypoint Mobility
SIFS	Short Inter Frame Space
GF	Group Security Failure
SMP	Semi-Markov Process
SPN	Stochastic Petri Net
SPNP	Stochastic Petri Net Package
STR	Skinny Tree
ULT	Untrusted Leave Threshold-based
SMPL	Simulation Modeling Programming Language
TAUDT	Trusted And Untrusted Double Threshold-based
TBR	Threshold-based Periodic Batch Rekeying
TGDH	Tree-based Group Diffie-Hellman
UAV	Unmanned Aerial Vehicle
VRT	Variance Reduction Technique

Appendix B: Summary of Notation

λ	Arrival rate of join requests (times/s)
μ	Arrival rate of leave requests (times/s)
P_t	Probability of trustworthiness, i.e., probability that a user request is issued from a trusted user
T_b	Average overhead (delay) for broadcasting in the wireless network (Chapter 4)
BW	Wireless network bandwidth (Mbps)
C_m	Communication overhead bits in each rekeying state
S_{cm}	Average communication overhead (delay) per rekeying operation
S	Average communication overhead (delay) per join/leave operation
P_v	Average probability of secrecy violation
D	Average delay occurred per join/leave operation
T	Average batch rekey interval
N	Total number of members in the group (Chapter 4)
J	Length of each key (bits) in LKH
a	Number of trusted join requests
b	Number of trusted leave requests
c	Number of untrusted leave requests
ULT	Untrusted Leave Threshold-based: This protocol has only one threshold, *k3*, to check against the number of untrusted leave requests (i.e., *c* in the state representation)
TAUDT	Trusted and Untrusted Double Threshold-based: This protocol has two thresholds, *k1* and *k2*, with *k1* checking against the number of trusted requests (i.e., $a+b$) and *k2* checking against the number of untrusted leave requests (i.e., *c*).
JALDT	Join and Leave Double Threshold-based: This protocol has two thresholds, *k1* and *k2*, with *k1* checking against the number of trusted join requests (i.e., *a*) and *k2* checking against the number of leave requests (i.e., $b+c$)
k1	First threshold used by *TAUDT* and *JALDT*
k2	Second threshold used by *TAUDT* and *JALDT*
k3	Only threshold used by *ULT*

T_{IDS}	An initial constant intrusion detection interval in the intrusion detection function (i.e., $D\ (m_d)$)
λ_c	An initial constant rate used in the attacker function (i.e., $A\ (m_c)$)
$D\ (m_d)$	Detection function that returns a periodic detection rate based on m_d
m_d	Degree of compromised nodes that have been detected by IDS
$A\ (m_c)$	Attacker function that returns time taken to compromise a node based on m_c
m_c	Degree of compromised nodes currently in the system
λ_q	Group data communication rate per node (times/s)
$p1$	False negative probability of host-based IDS
$p2$	False positive probability of host-based IDS
T_{cm}	Communication time for broadcasting a rekeying message (s)
b_{GDH}	Length of an intermediate value in applying GDH.3 (bits)
b_{GC}	Packet size for normal group communication activities (bits)
b_S	Packet size for status exchange (bits)
b_v	Packet size for a vote or a final decision (bits)
b_{ack}	Packet size for an acknowledgement (bits)
p	A base or exponent used in $D\ (m_d)$ and $A\ (m_c)$
T_{status}	Periodic time interval for transmitting a status exchange packet (s)
M_{vote}	Number of actual vote-participants against a target node
m	Number of initial vote-participants against a target node
N_{init}	Initial number of trusted member nodes in the system
N	Number of current trusted member nodes (Chapter 5)
$MTTSF$	Mean time to security failure (s)
$\hat{C}_{total}$	Total communication cost incurred due to all activities per time unit (bits/s)
$\bar{R}$	Average service response time per group communication operation (s)
$\hat{C}_{GC}$	Group communication cost (bits/s)
$\hat{C}_{rekey}$	Communication cost incurred by rekeying operations due to join/leave and eviction events (bits/s)
$\hat{C}_{status}$	Communication cost incurred by status exchange periodically (bits/s)
$\hat{C}_{IDS}$	Communication cost incurred by intrusion detection activities (bits/s)
C_{GDH}	Communication cost incurred per rekeying operation using GDH.3 (bits/s)

Λ_{J+L}	Aggregate group join and leave rate in equilibrium (times/s)
T_{RTS}	Transmission delay for RTS (request-to-send) (s)
T_{CTS}	Transmission delay for CTS (clear-to-send) (s)
$SIFS$	Short inter-frame space (s)
$DIFS$	Distributed inter-frame space (s)
T_{slot}	Slot time of contention-window size to calculate random backoff period in DCF (distributed coordination function) (s)
$E[CW]$	Average contention-window size
T_{com}	Transmission delay for a packet (s)
T_b	Wireless network delay including channel contention time (s) (Chapter 5)
T_c	Channel contention delay with an idle channel (s)
T_{off}	Channel contention delay due to random backoff period when the channel is not idle (s)
Q	Success packet transmission probability without collision occurred
λ_{packet}	Packet arrival rate (times/s)
K_G	Group key
K_{RL}	Leader key
K_{Ri}	Regional key in region *i*
RV_i	Regional view in region *i*
LV	Leader view
GV	Group view
RL_i	A leader in region *i*
$RM_{j,\,i}$	A member *j* in region *i*
σ	Mobility rate per node
$\lambda_{np,i}$	Per-group group partitioning rate when the number of groups = *i*
$\mu_{nm,i}$	Per-group group merging rate when the number of groups = *i*
λ_p	Node density (nodes/km^2)
λ_q	Group communication rate per node
A	Operational area of the mobile group, that is, $A = \pi r^2$ (km^2) where r is the radius
A_{region}	Area of a region
s	Radius of a hexagon region

r	Radius of the operational area
R	Wireless per-hop radio range (*m*)
k	Size of a group key (*bits*)
v	Size of each intermediate value in GDH (*bits*)
T_{RB}	Intra-regional beaconing interval (*s*)
T_{LB}	Inter-regional beaconing interval (*s*)
M_{alive}	Size of a beaconing message (*bits*)
M_q	Size of a group communication message (*bits*)
U_{view}	Size of an update message (*bits*)
$C_{np,\,i}$	Cost per group partition event when the number of groups = *i* (*hop bits/s*)
$C_{nm,\,i}$	Cost per group merge event when the number of groups = *i* (*hop bits/s*)
C_{intra}	Cost for intra-key rekeying and regional view updating in a region (*hop bits/s*)
$C_{inter,\,i}$	Cost for inter-key rekeying and leader view updating in a group (*hop bits/s*)
H_{region}	Number of hops between a leader and a member within a region
$H_{leader,\,i}$	Number of hops among leaders in a group when the number of groups = *i*
$R(n)$	Number of regions in the system
$N_{region\,,i}$	Number of regions in a group when the number of groups = *i*
N_{np}^{region}	Number of partitioned regions in a group after a group partition event
N_{nm}^{region}	Number of merged regions in a group after a group merge event
$N_{region}^{members}$	Number of members in a region
C_{rekey}^{intra}	Intra-regional cost for rekeying a regional key (*hop bits/s*)
C_{update}^{intra}	Intra-regional cost for updating a regional view (*hop bits/s*)
$C_{rekey\,,i}^{inter}$	Inter-regional cost for rekeying a leader key when the number of groups = *i* (*hop bits/s*)
$C_{update\,,i}^{inter}$	Inter-regional cost for updating a leader view when the number of groups = *i* (*hop bits/s*)
$C_{leader\,,i}^{change}$	Cost for a leader change in a group when the number of groups = *i* (*hop bits/s*)
σ_n	Regional mobility rate per node
Λ_J	Aggregate group join rate
Λ_L	Aggregate group leave rate
Λ_{RB}	Aggregate periodic beaconing rate from all members in the system

Λ_{LB}	Aggregate periodic beaconing rate from all leaders in the system
Λ_q	Aggregate group communication rate from all members in the system