ABSTRACT

ADAPTIVE TRUST AND REPUTATION SYSTEM AS A SECURITY SERVICE IN GROUP COMMUNICATIONS

by
Pitipatana Sakarindr

Group communications has been facilitating many emerging applications which require packet delivery from one or more sender(s) to multiple receivers. Owing to the multicasting and broadcasting nature, group communications are susceptible to various kinds of attacks. Though a number of proposals have been reported to secure group communications, provisioning security in group communications remains a critical and challenging issue.

This work first presents a survey on recent advances in security requirements and services in group communications in wireless and wired networks, and discusses challenges in designing secure group communications in these networks. Effective security services to secure group communications are then proposed. This dissertation also introduces the taxonomy of security services, which can be applied to secure group communications, and evaluates existing secure group communications schemes.

This dissertation work analyzes a number of vulnerabilities against trust and reputation systems, and proposes a threat model to predict attack behaviors. This work also considers scenarios in which multiple attacking agents actively and collaboratively attack the whole network as well as a specific individual node. The behaviors may be related to both performance issues and security issues. Finally, this work extensively examines and substantiates the security of the proposed trust and reputation system.

This work next discusses the proposed trust and reputation system for an anonymous network, referred to as the Adaptive Trust-based Anonymous Network (ATAN). The distributed and decentralized network management in ATAN does not require a central authority so that ATAN alleviates the problem of a single point of failure. In ATAN, the trust and reputation system aims to enhance anonymity by establishing a trust and reputation relationship between the source and the forwarding members. The trust and reputation relationship of any two nodes is adaptive to new information learned by these two nodes or recommended from other trust nodes. Therefore, packets are anonymously routed from the 'trusted' source to the destination through 'trusted' intermediate nodes, thereby improving anonymity of communications. In the performance analysis, the ratio of the ATAN header and data payload is around 0.1, which is relatively small.

This dissertation offers analysis on security services on group communications. It illustrates that these security services are needed to incorporate with each other such that group communications can be secure. Furthermore, the adaptive trust and reputation system is proposed to integrate the concept of trust and reputation into communications. Although deploying the trust and reputation system incurs some overheads in terms of storage spaces, bandwidth and computation cycles, it shows a very promising performance that enhance users' confidence in using group communications, and concludes that the trust and reputation system should be deployed as another layer of security services to protect group communications against malicious adversaries and attacks.

ADAPTIVE TRUST AND REPUTATION SYSTEM AS A SECURITY SERVICE IN GROUP COMMUNICATIONS

by
Pitipatana Sakarindr

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering

Department of Electrical and Computer Engineering

January 2010

APPROVAL PAGE

ADAPTIVE TRUST AND REPUTATION SYSTEM AS A SECURITY SERVICE IN GROUP COMMUNICATIONS

Pitipatana Sakarindr

10/21/09

Dr. Nirwan Ansari, Dissertation Advisor — Date
Professor of Electrical and Computer Engineering, NJIT

10/21/09

Dr. Roberto Rojas-Cessa, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

10/21/09

Dr. Edwin Hou, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

10/21/09

Dr. Yanchao Zhang, Committee Member — Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Oct 21, 09

Dr. Rajarathnam Chandramouli, Committee Member — Date
Thomas E. Hattrick Chair Professor of Information Systems
Electrical and Computer Engineering, Stevens Institute of Technology

BIOGRAPHICAL SKETCH

Author: Pitipatana Sakarindr

Degree: Doctor of Philosophy

Date: January 2010

Undergraduate and Graduate Education:

- Doctor of Philosophy in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2010

- Master of Science in Computer Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2002

- Bachelor of Science in Electrical Engineering,
  King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand, 1999

Major: Electrical Engineering

Presentations and Publications:

Pitipatana Sakarindr, Chao Zhang, and Nirwan Ansari,
"Adaptive Trustworthiness-based Enhancements in AODV Protocol for Wireless Ad Hoc Networks,"
in preparation.

Pitipatana Sakarindr, Chao Zhang, and Nirwan Ansari,
"Investigating Trust and Reputation Systems and Security Analysis in Communications Networks,"
in preparation.

Pitipatana Sakarindr and Nirwan Ansari,
"Security Services on Group Communications,"
IET Information Security Special Issue on Multi-Agent and Distributed Information Security, June 2010.

Nirwan Ansari, Pitipatana Sakarindr, Ehsan Haghani, Chao Zhang, Aridaman K. Jain, Yun Q. Shi,
"Evaluating Electric Voting Systems Equipped with Voter-Verified Paper Records,"
IEEE Security and Privacy, vol. 6, no. 3, pp. 30-39, May 2008.

Nirwan Ansari, Chao Zhang, Roberto Rojas-Cessa, Pitipatana Sakarindr, and Edwin Hou,
"Networking for Critical Conditions,"
IEEE Wireless Communication Magazine, vol. 15, no. 2, pp. 73-81, January 2008.

Pitipatana Sakarindr and Nirwan Ansari,
"Adaptive Trust-based Anonymous Network,"
International Journal of Security and Networks (IJSN), Special Issue on Computer and Network Security, vol. 2, no. 1/2, pp. 11-26, 2007.

Pitipatana Sakarindr and Nirwan Ansari,
"Security Services in Group Communications over Wireless Infrastructure, Mobile Ad-Hoc, and Wireless Sensor Network,"
IEEE Wireless Communication Magazine, Special Issue on Security in Wireless Mobile Ad Hoc and Sensor Networks, vol. 14, no. 5, pp. 8-20, October 2007.

Pitipatana Sakarindr and Nirwan Ansari,
"Elliptic Curve Cryptosystem-based Group Key Management on Secure Group Communications,"
Proceedings of the 2007 IEEE Conference on Military Communications, Orlando, Florida, pp. 1-6, October 2007.

Pitipatana Sakarindr, Nirwan Ansari, Roberto Rojas-Cessa, and Symeon Papavassiliou,
"Information Assurance in the SQoS Network,"
Proceedings of the 2006 IEEE Sarnoff Symposium, Princeton, New Jersey, pp. 1-4, March 2006.

Yanquio Luo, Pitipatana Sakarindr, and Nirwan Ansari,
"On the Survivability of WDM Optical Networks", *e-Business and Telecommunication Networks 2006*, Netherlands: Springer, 2006, pp: 31- 40.

Pitipatana Sakarindr, Nirwan Ansari, and Roberto Rojas-Cessa,
"Security-enhanced Quality of Service: A Network Analysis,"
Proceedings of the 2005 IEEE Conference on Military Communications, Atlantic City, New Jersey, vol. 4, pp. 2165-2171, October 2005.

Pitipatana Sakarindr, Nirwan Ansari, Roberto Rojas-Cessa, and Symeon Papavassiliou,
"Security-enhanced Quality of Service: Design and Architecture,"
Proceedings of the 2005 IEEE Sarnoff Symposium, Princeton, New Jersey, pp. 129-132, April 2005.

To my beloved ones for their love, support, and encouragement,
I cherish from the bottom of my heart.

ACKNOWLEDGMENT

First and foremost, I would like to express my deepest appreciation to my advisor, Dr. Nirwan Ansari, for providing me with most valuable and prodigious resources, guidance, opportunities, and heartfelt support all through my graduate education in both the masters and doctoral programs. He has always been a source of encouragement and knowledge. His perseverance, sincerity and hardworking qualities will always serve as an inspiration to me.

I would also like to gratefully thank Dr. Roberto Rojas-Cessa, Dr. Edwin Hou, Dr. Yanchao Zhang and Dr. Rajarathnam Chandramouli, who were kind enough to actively participate in my dissertation committee, providing constructive comments and valuable suggestions at all stages of this work.

I also express my deep gratitude to my past and current colleagues at the Advanced Networking Laboratory, NJIT, who supported me in many ways and provided me suggestions over the years, especially Yuanqiu Luo, Wei Yan, Kai Xu, Ye Tian, Gang Cheng, Chao Zhang, Amey Shevtekar, Nan Wang, Ehsan Haghani, Zhen Qin, Jingjing Zhang, Si Yin, Haijun Pan, and Apoorv Khare. I cherish and will always remember their friendships.

Finally, no words are enough to acknowledge the constant support and encouragement of my family, especially my parents, my sisters (Tee, Tue, Tom, Took), Aunt Chantana Somboontum and Uncle Vince Rosati, my brothers in law, Fabrizio Bivona and Peeriyathep Homhuan, and Naruemon Suwattananont. Thank you.

TABLE OF CONTENTS

Chapter	Page
1 INTRODUCTION	1
1.1 Background Information	1
1.2 Objectives	1
1.3 Organization	2
2 SECURITY SERVICES ON GROUP COMMUNICATIONS	4
2.1 Objective	4
2.2 Introduction	4
2.3 Known Attacks in Group Communication Systems (GCSs)	6
2.4 Security Requirements for Group Communication Systems	9
2.5 Performance Attributes to Evaluate Secure Group Communication Systems	11
2.5.1 Fundamental Attributes	11
2.5.2 Service-specified Attributes	15
2.6 Security Services for Group Communication Systems	20
2.6.1 Group Key Management (GKM)	21
2.6.2 Group Access Control	23
2.6.3 Group Signature	24
2.6.4 Group Anonymity	25
2.6.5 Secure Routing	28
2.7 Group Communications-oriented Networks	29
2.7.1 Multi-Agent System	29

TABLE OF CONTENTS
(Continued)

Chapter **Page**

2.7.2 Personal Area Network 30

2.7.3 Multicast Security in IP Multicast Networks 32

2.8 Challenging Factors in Designing Secure GCSs 33

2.8.1 Environment and System Performance 33

2.8.2 Efficiency of Key Management and Distribution 34

2.8.3 Early Detection and Prevention 34

2.8.4 Increased Concern over Privacy 35

2.8.5 Implementation of Security Services for Different Applications 35

2.9 Conclusion 35

3 SECURITY SERVICES IN GROUP COMMUNICATIONS OVER WIRELESS INFRASTRUCTURE, MOBILE AD-HOC, AND WIRELESS SENSOR NETWORKS 39

3.1 Objective 39

3.2 Introduction 39

3.3 Known Attacks in Wireless Networks 40

3.3.1 Data Integrity and Confidentiality-related Attacks 41

3.3.2 Power Consumption –related Attacks 42

3.3.3 Service Availability and Bandwidth Consumption –related Attacks 43

3.3.4 Routing –related Attacks 44

3.3.5 Identity –related Attacks 45

3.3.6 Privacy –related Attacks 46

TABLE OF CONTENTS
(Continued)

Chapter	Page
3.4 Secure Group Communication Systems	47
3.5 Security Requirements and Security Services in SGC	49
3.6 SGC over Wireless Infrastructure Networks	53
3.7 SGC over Mobile Ad Hoc Networks (MANETS)	57
3.8 SGC over Wireless Sensor Networks	59
3.9 Open Challenges	66
3.10 Conclusion	67
4 ADAPTIVE TRUST AND REPUTATION SYSTEM	69
4.1 Objective	69
4.2 Introduction of Trust and Reputation System	69
4.3 The Adaptive Trust and Reputation System	70
4.3.1 Definitions of Trustworthiness, Trust, and Reputation	70
4.3.2 Trustworthiness Initialization Process	73
4.3.3 Trustworthiness Monitoring Process	75
4.3.4 Error Detection Process	77
4.3.5 Trustworthiness Request Process	79
4.3.6 Trustworthiness Re-evaluation Process	83
4.3.7 Trustworthiness Update Process	91
4.3.8 Forecasting	93
4.4 Threat Model against The Trust and Reputation System	95

TABLE OF CONTENTS
(Continued)

Chapter **Page**

4.4.1 Classifications of Attacks against the Trust and Reputation System 95

4.4.2 Attacks on the Trust and Reputation System 98

4.4.3 Attacks on the Information system 100

4.4.4 Attacks on Other Protocols or Applications 100

5 ELLIPTIC CURVE CRYPTOSYSTEM-BASED GROUP KEY MANAGEMENT FOR SECURE GROUP COMMUNICATIONS 102

5.1 Objective 102

5.2 Introduction 102

5.3 Background Information 104

5.3.1 Elliptic Curve 104

5.3.2 Elliptic Curve Cryptosystem (ECC) 106

5.3.3 Key Management Schemes in Group Communications 107

5.3.4 Cluster Based Group Key Management 109

5.4 ECC-based Key Management Scheme 110

5.4.1 The Cluster Key Establishment 111

5.4.2 The Group Key Establishment 112

5.4.3 Individual Join 114

5.4.4 Individual Departure 118

5.4.5 Cluster Head Departure 120

5.4.6 Periodic Rekeying 121

TABLE OF CONTENTS
(Continued)

Chapter	Page
5.5 ECC-based GKM Analysis	121
5.6 Conclusion	123
6 ADAPTIVE TRUST-BASED ANONYMOUS NETWORK	124
6.1 Objective	124
6.2 Introduction	124
6.3 Background Information	125
6.3.1 Anonymous Networks	125
6.3.2 Diffie-Hellman Cryptosystem	126
6.4 ATAN Framework	128
6.4.1 Terminologies, Definitions, Expressions, and Assumptions	129
6.4.2 Cluster-based ATAN Network Management	132
6.4.3 Trust and Reputation System	144
6.4.4 Transmission Processes	154
6.5 Threat Models	169
6.6 Network Analysis	172
6.7 Conclusion	173
7 CONCLUSION AND FUTURE WORK	174
APPENDIX THE PROBABILITY MODEL	176
REFERENCES	182

LIST OF TABLES

Table		Page
2.1	Some Known Attacks on Group Communications Based on Three Attack Attributes	7
2.2	The Comparison Table of Secure GCSs Along With Fundamental Performance-evaluating Attributes	37
2.3	The Comparison Table of Secure GCSs Along With Service-specified Performance-evaluating Attributes	38
3.1	Characteristics of Possible Attacks on SGC over Wireless Networks	48
3.2	Security Services to Countermeasure Attacks	53
3.3	Characteristics of Security Services in SGC over Wireless Networks	65
3.4	Comparison of SGC over Wireless Networks	68
4.1	Re-evaluation Scenarios	84
5.1	The Point Addition Properties	105
5.2	The Performance of Key Establishment in the ECC-GKM Protocol	122
6.1	Information Pattern for Error Packets	131
6.2	The Exemplary ATR Database of Node $N(I,1)$	145
6.3	The Sample Set of Criteria and Scores	166

LIST OF FIGURES

Figure		Page
2.1	Security services in provisioning secure group communications	6
2.2	Fundamental attributes for evaluating performance and security in a SGC system	14
2.3	Service-specified attributes for evaluating security services in a SGC system	15
3.1	An illustration of mixed attacks in a real wireless network	41
3.2	An illustration of the routing-related attacks and other attacks	47
4.1	An illustration of the error detection process	78
4.2	An illustration of the trustworthiness request process	82
4.3	An illustration of the trustworthiness reevaluation process	91
4.4	An illustration of the trustworthiness update process	93
4.5	A classification of attacks	101
5.1	An illustration of the ECC-based key exchange	107
5.2	An illustration of the key secrecy	108
5.3	Two levels of communications	109
5.4	A cluster key establishment	111
5.5	A group key establishment	113
5.6	A cluster key rekeying operation when the new member joins in	115
5.7	A group key rekeying operation when the new member joins in	118
6.1	An illustration of the Diffie-Hellman cryptosystem	126
6.2	An illustration of the ElGamal cryptosystem	127
6.3	An illustration of communication levels in ATAN	133

LIST OF FIGURES
(Continued)

Figure		Page
6.4	An illustration of the JOIN procedure	138
6.5	An illustration of various processes for data transmission in ATAN	147
6.6	The packet format in ATAN	155
6.7	The forwarding steps in the probing process	161

CHAPTER 1

INTRODUCTION

1.1 Background Information

Group communications refers to either point-to-multipoint or multipoint-to-multipoint communications via some underlying networking infrastructure such as VPN and IP multicast networks. Group communications in wired networks has been facilitating many emerging applications which require packet delivery from one or more sender(s) to multiple receivers. There are increasingly high demands of security on group communications such as authentication, authorization, and privacy. Though a number of proposals on secure group communication systems (GCSs) have been reported, provisioning security in group communications remains a critical and challenging networking issue. Owing to insecure wireless channels, group communications are susceptible to various kinds of attacks. Though a number of proposals have been reported to secure group communications, provisioning security in group communications in wireless networks remains a critical and challenging issue.

1.2 Objectives

Two objectives of this dissertation are to expand the utilization of elliptic curve cryptosystem (ECC) into GKM to make the key management in GKM more efficient, and to deploy trust and reputation to enhance security services in group communications systems.

1.3 Organization

This dissertation is comprised of three parts. The first part includes Chapters 2 and 3. Chapter 2 presents a survey on recent advances in security requirements and services and challenges in many group communications systems [1]. Chapter 3 presents a survey on recent advances in security requirements and services and challenges in three different wireless network types: wireless infrastructure networks, mobile ad hoc networks, and wireless sensor networks [2]. Chapter 4 introduces the trust and reputation system with two objectives: to propose the resilient trust and reputation system that can be implemented in different network environments; to analyze security concerns in many aspects including an attack classification, an illustration of attack scenarios, and an introduction of defense mechanisms responding dependently on every attack scenarios.

The second part expands the utilization of elliptic curve cryptosystem (ECC) into GKM to decrease the key length while providing securities at the same level as that of other cryptosystems, and proposes the cluster scheme to make the key management more efficient, which is discussed in Chapter 5 [3]. This chapter utilizes the elliptic curve cryptosystem for group key management such that the group key is securely selected, rekeyed, and distributed in group communications with a shorter key length but provides the same security level as that of other cryptosystems. Afterward, this chapter deploys the cluster scheme to decrease the processing time and key computations.

The third part presents a novel adaptive trust-based anonymous network (ATAN), which is discussed in Chapter 6 [4]. In ATAN, the trust and reputation model aims to enhance anonymity and security in GCS by establishing a trust and reputation

relationship between the source and the forwarding members. The trust and reputation relationship is adaptive to new information learned by these two nodes or recommended from other trust nodes. Therefore, packets are anonymously routed from the "trusted" source to the destination through "trusted" intermediate nodes. Finally, Chapter 7 draws conclusive remarks and discusses future work.

CHAPTER 2

SECURITY SERVICES ON GROUP COMMUNICATIONS

2.1 Objective

Securing group communications has attracted much attention as group-oriented communications has been increasingly facilitating many emerging applications which require packet delivery from one or more sender(s) to multiple receivers. Overall, security remains a critical and challenging networking issue in group communications. Different security services may be needed to satisfy different security requirements from various applications. The most fundamental component of security services is the cryptographic material, such as keys. Of all proposals reported, most have focused on addressing the issue of key management to secure group communication systems. However, any secure group communications system should offer as many security services as they can. This chapter presents a survey on recent advances in several security requirements and security services in many group communications systems, and challenges in designing a secure group communications system.

2.2 Introduction

Group communications refers to either point-to-multipoint or multipoint-to-multipoint communications via some underlying networking infrastructures. This chapter does not specify the underlying networking infrastructures to support group communications systems (GCSs) since there are currently no concrete works or standards on those networks to effectively secure group communications. However, the ongoing activities of

the IETF Multicast Security Charter work group (IETF MSEC WG) standardize the multicast security framework and architectures for IP-based multicast networks.

Owing to the distributed nature of group communications, it is difficult to ensure the basic security desires: only authenticated senders can transmit authenticated and protected message; only authorized receivers receive a meaningful message; and all activities can be partially or fully accounted for. Different security services may be needed to satisfy different security requirements for different applications. The most fundamental component of security services is the cryptographic material, such as keys. Inherently, the performance of security services fundamentally relies on the strength and security of the cryptographic material. Many proposals developed so far to secure group communication systems have focused on solving the issue of key management. However, any secure group communications system should offer as much security services as they could. This chapter provides extensive investigation into six security services, including group key management, group access control, group anonymity, group signature, and secures routing, so that readers can understand broader issues on secure group communications, as depicted in Figure 2.1.

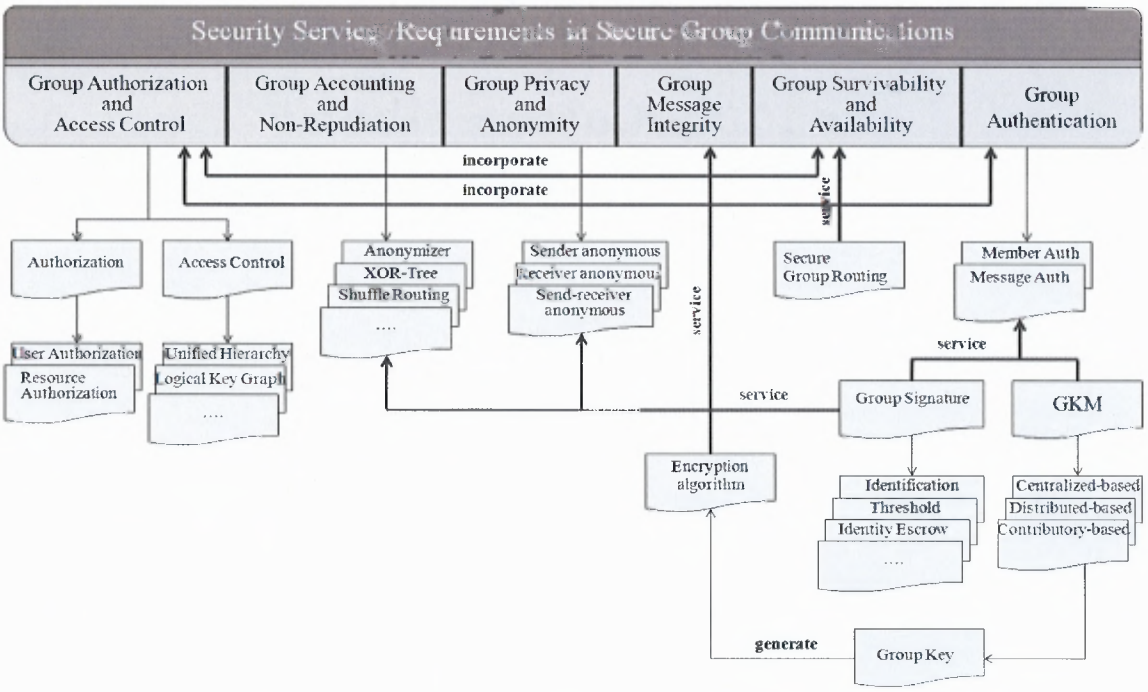


Figure 2.1 Security services in provisioning secure group communications [1].

The rest of this chapter is organized as follows. First, security requirements in group communications are discussed, followed by various known attacks targeting group communication systems. Then, corresponding security services proposed in some outstanding GCSs along with attributes for evaluating each service and a comparison of these attributes are presented. Some existing group-oriented applications that are deployed in several networks are later illustrated. Finally, the future challenges and the summary of this chapter are discussed.

2.3 Known Attacks in Group Communication Systems (GCSs)

Some attacks have been discussed in many researches on providing security to group-oriented systems [5]. Some of most known attacks are classified based on three attack perspectives: whether the attack is passive or active; whether the attack targets on data or

control messages; and whether the attack mainly aims at group members or group controllers. The passive attacker passively intercepts messages or keeps track of communications. The active attacker modifies, injects or drops messages.

There are some known attacks that effectively compromise security of group communications as illustrated according to the attack attributes depicted in Table 2.1.

Table 2.1 Some Known Attacks on Group Communications Based on Three Attack Attributes [1]

Known Attacks \ Attack Attributes	Type of attacks <Passive/Active>	Type of messages being targeted at <Data/Control>	Type of affected victims <Member/Authority>	Associated security requirements
Denial of service	active	data	member and authority	availability and access control
Collusion	active and passive	data and control	Member and authority	possibly all requirements
Traffic analysis	passive	data	member	anonymity and privacy
Replay	active	data and control	member and authority	availability and access control
Cut-and-paste	active	data	member	authentication and message integrity
Loop routing	active	control	authority	availability and survivability
Blackhole routing	active	control	authority	availability and survivability
False or unauthorized routing update	active	control	authority	authentication, availability and access control
Impersonate	active and passive	data and control	member	authentication, access control, non-repudiation, and anonymity
Byzantine	active	data	member	authentication and message integrity
Member self-promotion	active	control	member	authentication
Eavesdropping	passive	data and control	member and authority	message integrity and confidentiality
Single point of failure	active	control	authority	availability and survivability

Denial of Service attack – An attacker sends out a large number of packets to exhaust resources and operations at the multicast routers or group controlling entities such as a key server and a group manager. The attacker may join in the multicast group and later interrupts the group operations, e.g., joining and rekeying processes. This results in denial of service to other users.

Collusion attack – A group of attackers can collude to attack the victim or to collect and exchange group-related information in order to instigate other attacks on the

victim, such as severe group exploitation, a masquerade of a legitimate member, and an illegal transmission of messages on behalf of other members.

Traffic analysis attack – An attacker correlates all incoming and outgoing packets at a forwarding entity to execute the traffic analysis on messages. The analysis may examine the message length, the message pattern or coding, and the duration of the message stayed in each group entity. The attacker may identify a possible sender or receiver, or determine the sending and receiving ends of communications.

Replay attack – An attacker repeatedly sends a large number of packets, which have been previously transmitted, to consume tremendous amount of bandwidth or to exhaust the victim's queues, resulting in dropping of other messages.

Cut-and-Paste attack – An attacker replaces the whole portion of encrypted data with its own false data, but leaves other non-encrypted portions (e.g., header portions) untouched. Thus, the attacker does not need to possess the decryption key but is still able to fabricate the message.

Routing attack – An attacker can execute attacks in various means such as targeting the routing infrastructures, exploiting the routing protocols, and fabricating the routing update messages. The consequences could be as follows: some multicast routers are isolated; group messages are routed in loop and then dropped after the Time-To-Live (TTL) expires; group messages are falsely forwarded to unauthorized attackers; blackhole routing; and an unauthorized addition into the routing table.

Impersonate attack – Regardless of how an attacker obtains the victim's identity, the attacker impersonates that group member to launch some attacks or carries out a theft of service on behalf of the victim.

Byzantine attack – An attacker is perverted and sends out multiple group messages to different subsets of other group members but maliciously claims that these messages are the same. Thus, the victim creates the wrong message sequence which may allow the attacker to corrupt the victim's machine or to create a backdoor in the victim's machine for future control.

Member serialization attack – Most contributory group key agreement protocols generate the shared group key in a serial order. An attacker targets a single participating member and effectively disrupts the key generation and rekeying processes.

2.4 Security Requirements for Group Communication Systems

This section describes commonly known security requirements for group communications, and security services that meet these requirements are later elicited in the subsequent section.

Group authentication: It enables a group member to be authenticated as unspecified, but legitimate such that the sending member can multicast a message on behalf of the group without revealing its identity during the verification process performed by the receiver. Besides user authentication, message authentication allows any group message to be verified of its authenticity.

Group authorization and access control: Every member may be assigned the same or different permissions and restrictions for accessing group resources. The access-controlling entity can verify a member's request to access specified resources by using several means such as the access control list (ACL) and access hierarchy.

Group accountability and non-repudiation: All group operations should be accountable, implying that any group operation performed and resources utilized can be

tracked and recorded in order to detect any abusing usages of resources and operations. A non-repudiation requirement ensures that the identity of a member whose activities are in dispute can be fully and precisely identified by the designated entity.

Group privacy and anonymity: Fundamentally, group privacy and anonymity contradict to group accountability and non-repudiation because the privacy of a malicious group member should be stripped off and its identity should be exposed. There have been some researches trying to determine the trade-offs between these requirements. For example, some threshold sharing mechanisms may allow a number of designated entities to gather information and to re-create some secret elements used to ultimately identify the wrong-doing members.

Group message integrity and confidentiality: Message integrity should be preserved by ensuring that the message has not been added or deleted or modified by any unauthorized entity, either unauthorized members or outsiders. In GCSs, the integrity is ensured by encrypting a group message with a single shared key, called a group key. Thus, the message protection mainly relies on the cryptographic strength of the group key. Confidentiality ensures that only the authorized can retrieve meaningful data from the message.

Group Survivability and Availability: An attacker may attack multicast routers and other routing infrastructures or target a joining operation in order to cut off some or all group members or disrupt group communications, causing service unavailability. To achieve group survivability, the routing protocol should ensure that any member can still be connected even under attacks. Furthermore, there should be some preventive

mechanisms to support group survivability by rediscovering connections in the events of link or node failures.

2.5 Performance Attributes to Evaluate Secure Group Communication Systems

In order to evaluate and compare different SGC systems, one needs to construct evaluation attributes to fairly analyze and determine the performance and security analysis of each SGC system. In this chapter, evaluation attributes are listed and grouped into two types: fundamental attributes used to evaluate mechanisms in providing one or more security services to GCSs; and specific attributes used as additional properties corresponding to those supported security services.

2.5.1 Fundamental Attributes

The fundamental attributes for a SGC system include the following as depicted in Figure 2.2.

- Type of group management. The group may be established and managed by three approaches: centralized (with a central authority), partially distributed (with a group of designated controllers), and fully distributed (without any explicitly designated controller). The group controller may perform the group initiation and termination, the membership admission, the group material generation, and the distribution of some controlling messages. The group controller may also act as a key server, if given the capacity.
- Overheads. In general, three types of overheads are incurred by all network operations: storage, communications, and processing. For storage overheads, a group controller and a group member may require different spaces of memory to

store group information such as session and group keys, list of group members, cryptography materials, and other service-related materials. For communications overheads, the characteristics of group communications likely incur additional communications messages. For example, dynamic group membership changes cause members to reorganize group operations (i.e., sending joining or leaving notifications, selecting new group controllers) and to rekey all related keys to ensure key secrecy (i.e., distribution of new keys). To process overheads, each group operation requires computation which can be measured in terms of the number of processing steps (iterations), processing duration, and complexity bound. The key generation and distribution, rekeying, and message encryption/ decryption/ digestion/ signing processes are computationally expensive.

- Scalability. The performance should not be degraded drastically as the group size increases and should be linear with the group size when implemented with a small or large group of members. In addition, the scalability may be increased in many following scenarios: for example, a group is managed in a distributed approach (due to easiness to expand the group).
- Dynamic membership. Group communications systems should be able to handle a membership change (i.e., any individual member leaves or joins the group at any time) without significant system performance degradation. Some systems may treat group merging and partitioning the same way as a bulk of individual membership changes, and may thus suffer degraded performance when handling a large group of membership changes. Different networks may require different ways to handle the group membership changes. For examples, wireless ad hoc

networks may observe higher mobility of members (more frequent membership changes) while multicast wired networks may expect less or even fixed mobility (less frequent membership changes).

- Trust relationship. Some systems require a trusted third party such as the certificate-issuing authority and the key server, which can make the trusted third entity a point of attack. Some systems assume trust relations among group controllers or between a group controller and members, but ignore the need for additional security mechanisms to protect trust operations such as trust establishment and updating trust relations. The security and performance analysis should take into consideration of such assumptions and reflect the actual effects of using trust relations in a SGC system.
- Resilience. It is necessary to include a threat model and the security analysis in designing and evaluating a SGC system. To analyze the threat model and security analysis, both network-based attacks and service-related attacks should be considered. The network-based attacks are general attacks that explore the vulnerabilities of a network. The service-related attacks specifically target the security service mechanisms originally deployed to satisfy some security requirements. For example, a group signature satisfies privacy and authentication, but unintentionally leaves the SGC system with new vulnerabilities, such as weaknesses in the signature algorithm or erred source codes of software generating the signatures. The security analysis may be able to detect and prevent such vulnerabilities.

- Control channels. Some systems require some off-line communications channels, such as using a telephone. Other systems may require control channels that are usually assumed to be secure and not restricted to a limited bandwidth resource. The performance and security analysis should measure on-line impacts of the off-line channel.

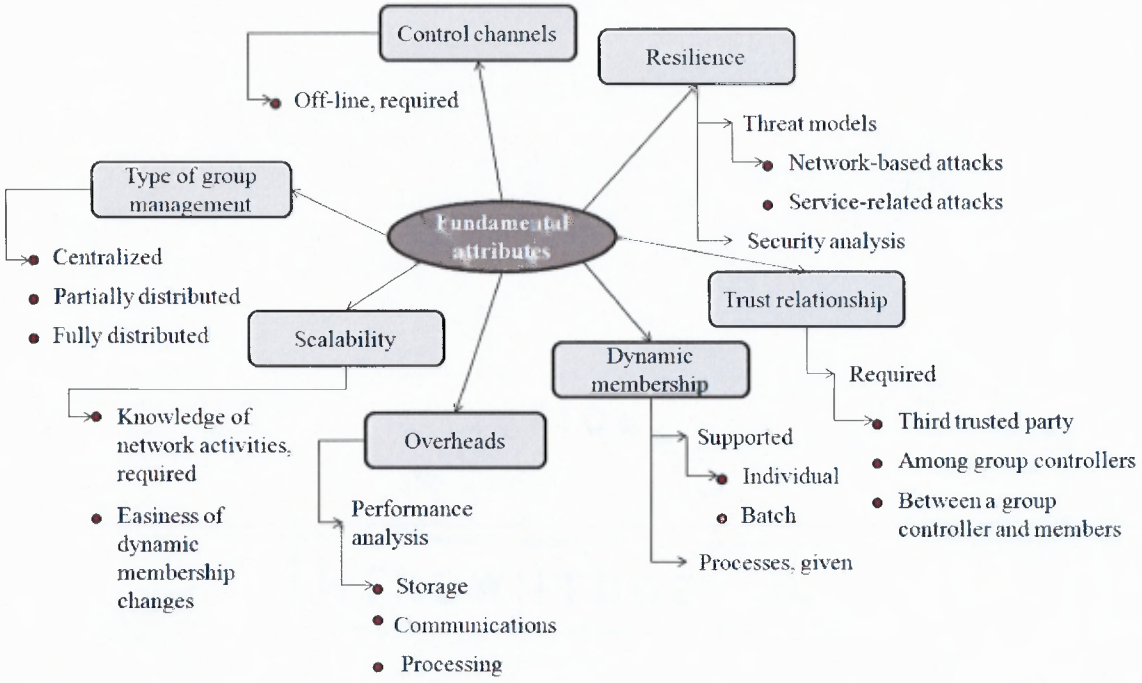


Figure 2.2 Fundamental attributes for evaluating performance and security in a SGC system [1].

2.5.2 Service-specified Attributes

Additional properties or attributes for specific security services may be discussed separately as follows as depicted in Figure 2.3.

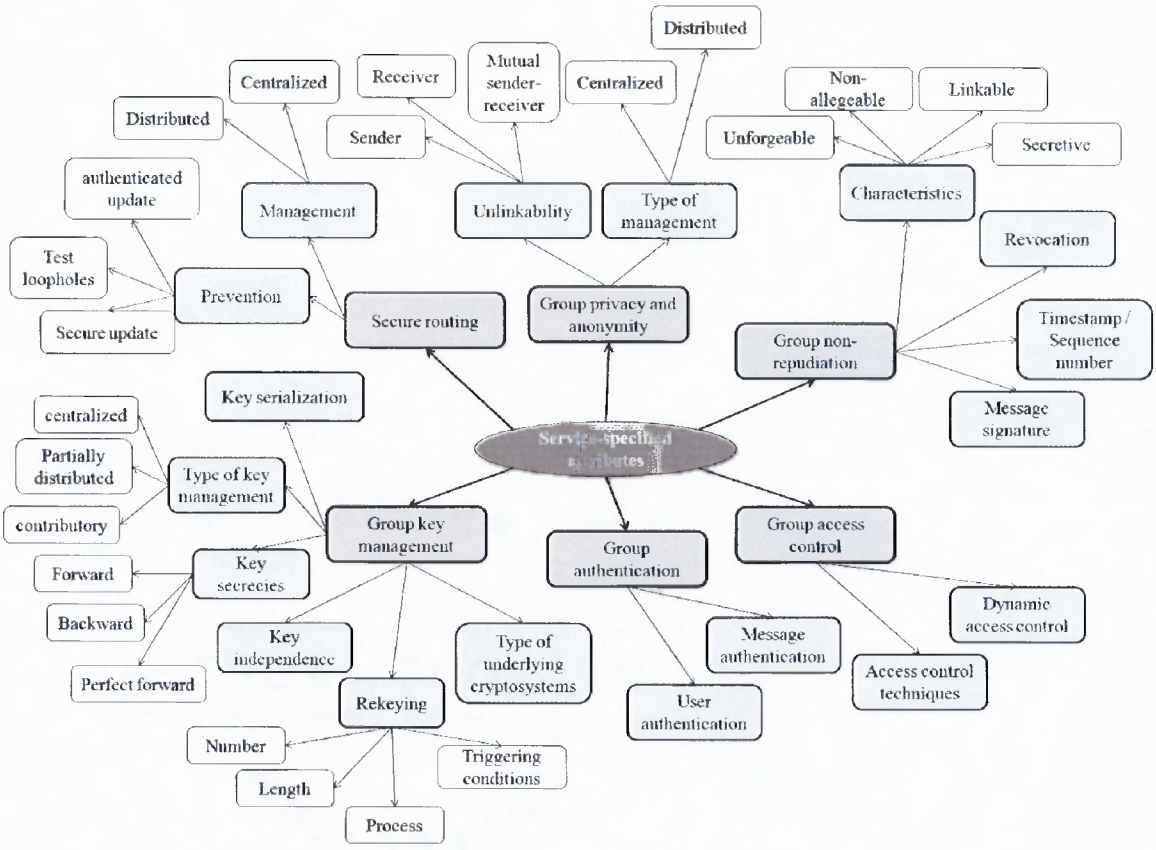


Figure 2.3 Service-specified attributes for evaluating security services in a SGC system [1].

1 Group Key Management. Six additional properties are related to a group key management scheme as follows.

- Type of key management. In a centralized key management, a key manager generates the keys, distributes them to associated members, and maintains all the keys. The security of key generation is strong, but the key manager carries most of the workload and becomes the attack target. In a partially distributed key

management, a set of key managers generate the keys and distribute them to all group members. Thus, each key manager has a reduced workload. Still, it is a point of target and the security of key generation is weakened. In a contributory key management, each member randomly selects its contribution, exchanges within a group, and generates a shared group key without a central key server/manager. The security of key selection and generation is low, but there is no need for a key manager. All members equally share the workloads.

- Type of underlying cryptosystems. Various cryptosystems can be deployed for encryption, decryption, and digestion, such as RSA, DL, and ECC, which determine the cryptographic strength of keys.
- Key secrecies. There are three aspects of key secrecies: forward secrecy, backward secrecy, and perfect forward secrecy. The forward secrecy ensures that a new joining member cannot use the new key to decrypt all messages which have been encrypted with the previous key(s). The backward secrecy ensures that a leaving member cannot use the previous key(s) to decrypt all messages encrypted with the new key. The perfect forward secrecy ensures that a compromise of a long-term key seed which generates the present short-term key(s) cannot deprive the secrecy of other previous short-term keys which have been generated by the compromised long-term key.
- Key independence. A disclosure of a subset of session keys cannot deprive the secrecy of other subsets of session keys which have been generated by the same long-term key seed.

- Key serialization. The key materials are selected and the group key is generated by members in an ordered sequence. An attack on any participating member disrupts the whole process. Instead, some schemes may construct the key by other means, i.e., broadcasting the key materials or establishing a key tree, at the expense of overheads.
- Rekeying. There are several factors in evaluating rekeying as follows. The number of rekey messages—the number of distributed and received messages per member or per key manager may be different. The length of rekey messages—some protocols aggregate multiple rekey messages into a single message, which in return increases the consumed bandwidth for one transmission. Thus, the performance analysis should also determine the bandwidth consumption per message in addition to the number of transmitted messages. The rekeying process—the rekeying operation should reduce or optimize the computation and time complexity of the rekeying operation with respect to the group size. Triggering conditions—there are three scenarios: first (membership changes based rekeying), keys associated with the membership changes must be rekeyed to ensure the key secrecy for the remaining members; second (periodic rekeying), the rekeying operation is invoked periodically to prevent keys from being compromised over time; and third (specified rekeying), a system enables the rekeying operation for specified incidents, such as upon detection of attacks or violations.

2 Group Authentication. Two additional properties are related to a group authentication scheme as follows.

- User authentication. Users should be authenticated upon joining the group, signing the messages, or accessing group materials.
- Message authentication. The system requires a sender to sign a message and a receiver to authenticate the received message for its authenticity and integrity.

3 Group Access Control. This chapter considers two additional properties related to a group access control scheme as follows.

- Access control. The group resources and group messages should be accessible only to authorized members.
- Dynamic access control. A system enables the member to dynamically change its request to access resources. Consequently, the system must be able to update access permissions and restrictions with additional mechanisms when the member's access privilege changes.

4 Group Non-repudiation. Three additional properties related to a group non-repudiation scheme are discussed as follows.

- Message signature. A system requires messages to be signed with a membership certificate to identify the originator (signer) of the message.
- Timestamp or sequence number. Timestamps and/or sequence numbers can be used to limit the validation of a certificate or message signature, and to prevent replay attacks.
- Revocation of certificates. The expired certificate or misuse of certificate is revoked by the issuer and may be publicly announced in the revocation list. Some

systems may keep the expired certificates for future verification at the expense of storage overhead.

- Characteristics of signature. There are four basic requirements of a digital signature: unforgeable—a group of colluded attackers cannot generate a group signature identical to that generated by a legitimate member; non-allegeable—a group of colluded attackers cannot generate a group signature by which a group controller falsely identifies a legitimate member as an attacker; linkable—a group of colluded attackers cannot generate a valid group signature by which a group controller cannot identify the identity of any of these attackers; secretive—a member's secret elements can neither be retrieved from a group signature nor from any part of it.

5 Group Privacy and Anonymity. Two additional properties related to a group privacy and anonymity scheme are discussed as follows.

- Unlinkability of anonymous communications. There are three anonymities: sender anonymity—a sender shall not be linked to its sent message to prevent attackers from learning of the message's origin; receiver anonymity—a receiver shall not be linked to the received message to prevent attackers from learning of the message's destination; and sender-receiver anonymity—the sender and receiver shall not be linked together and they are also relatively anonymous to each other.
- Type of management. There are two types in this subcategory: centralized management—a system relays messages through a trusted anonymous entity to hide identities of the sender and receiver; and distributed management—a system

relays messages through a group of anonymous entities or hides the identities by other means such as encapsulating messages and coding with the XOR operation.

6 Secure Routing. Two additional properties related to a secure routing scheme are considered.

- Management. A system can establish and maintain routing-related information in a centralized or distributed manner.
- Prevention. Updating routing information must be restricted to authorized members. A new routing path should be tested to prevent routing black hole and loopholes. Any request to join/add/update the routing table and other routing-related information should be authenticated and authorized.

Then, based on properties shown in Figures 2.2 and 2.3, the comparison results of these outstanding secure GCSs are presented in the following section, as summarized in Tables 2.2 and 2.3. The shaded box in Table 2.3 indicates that the security services are irrelevant to the systems and are excluded. Additionally, Table 2.3 notes that 1) N/A stands for "not applicable" or "no available information," and 2) if a system design does not offer some security services, the corresponding table cells of these security services are emptied and shaded.

2.6 Security Services for Group Communication Systems

This section discusses essential security services that meet security requirements mentioned before. Many concepts and existing solutions have been proposed to provide such services, but only a few promising concepts and solutions are highlighted here.

2.6.1 Group Key Management (GKM)

Any GKM scheme should exhibit the following properties: the key generation and rekeying should be provably secure; an imitation of the group key should be infeasible or computationally difficult; the group key is securely distributed and only the legitimate users can obtain a valid group key; and a revocation of the group key upon a membership change should be immediately notified.

Waldvogel *et al.* [6] proposed the VersaKey framework that organizes the key space to reduce the complexity. The framework consists of three approaches (centralized tree-based approach, and centralized and distributed flat table-based approaches) that support both centralized and fully distributed SGC environments. The centralized tree approach manages all keys by means of a hierarchical key tree. In flat-type approaches, the key tree is flattened into a table that stores all keys, each of which is indexed by some binary bits of the member's identification. In addition, the transitions among these approaches on-the-fly are presented. It was shown that the key management has a computational complexity of O(*logN*), where *N* is the group's size.

Banerjee and Bhattacharjee [7] proposed a management scheme based on a clustering protocol and a hierarchy of keys. All members are divided into several clusters in a layer. In each cluster, a cluster header will be selected and be a cluster member of the upper layer. This process is repeated until there is only one cluster member in the top layer. The clustering protocol is deployed to cluster the members in each layer such that when a membership changes, only one cluster in each layer requires its associated keys to be updated. It was demonstrated that, for an individual membership change, the overheads incurred by group members are constant with respect to the group size. In

addition, for a bulk membership change, the processing and communication overheads at the key server is logarithmic with respect to the group size.

Wong *et al.* [8] introduced three key graphs-based rekeying approaches (user-, key-, and group-oriented) to mitigate the scalability problem. In events of membership changes, three rekeying approaches operate as follows: for user-oriented rekeying, the key server generates new keys for each affected member and encrypts them with keys previously held by that member; for key-oriented rekeying, the new keys are encrypted individually with previous keys at the same key nodes of the key tree and multicast in multiple rekey messages; and for group-oriented rekeying, it is similar to the key-oriented rekeying except that all new keys are put together in a single rekey message. Simulation results demonstrated that the complexities of rekeying overheads of the three approaches are linear with the logarithm of the group size. In addition, the group-oriented approach performs the best from the perspective of the key server while the user-oriented approach has the best performance from the perspective of the group member.

Amir *et al.* [9] secured group communications with a secure service by using the proposed robust and contributory key agreement protocol and the virtual synchrony semantics. The proposed protocol enhances the group Diffie-Hellman (GDH) key agreement in two main incidents: first, it can mitigate the member serialization problem which requires the group key to be constructed or rekeyed in a serial ordering; second, it incorporates a membership protocol such that it is aware of any membership changes during the key generation and rekeying processes. In addition, the proposed protocol can effectively handle events of members joining and leaving within a very short time interval. Their simulated system, called Secure Spread, demonstrated the reduction of

time used to successfully establish a secure group and generate a group key after a membership change.

2.6.2 Group Access Control

In group-oriented networks, group members can be assigned with multiple access privileges. The data stream can be accessed with different access privileges such that only members who have an appropriate privilege can access corresponding portions of contents of the data stream (or some data streams of the aggregated data stream). This is referred to as multiple access privilege. In addition, some GCSs can support dynamic access control.

Sun and Liu [10] proposed a multi-group (MG) key management scheme to construct the logical key graph by integrating key trees of all members. Each authorized member holds a set of keys associated with the nodes from the leaf node to the root node in the key graph. The access privilege for each member is determined by the beheld set of keys. The scheme can provide forward and backward secrecies when a member changes its access privileges (or leaves the group) because the set of keys and resources associated with that member are re-assigned (or withdrawn). It was shown that overheads caused by the rekeying incidents are greatly reduced. In addition, the scalability and complexity of the scheme is improved.

Zhang and Wang [11] proposed a hierarchical access control (HAC) key management scheme, where a key server maintains the description of relations of memberships and resources in the form of unified hierarchy. Instead of classifying members with different resource requirements into multi groups as of the conventional multi-group (MG) key management scheme, the HAC scheme constructs a membership-

group subgraph and a resource-group subgraph, and combines them into a single unified logical key graph that determines which resource the specified member can access. Simulation results have demonstrated that, with the HAC scheme, the storage and rekeying overheads at every member and the key server can be significantly reduced by at least 20% as compared to those of the MG scheme.

2.6.3 Group Signature

The group signature is used to authenticate the source whether the message is sent from the signer who is a legitimate, but unidentified, group member and to authenticate the message whether it has been altered during transmission. In case of a dispute, the third trusted party or the group controller can identify the actual signer of the signed message.

Chen *et al.* [12] proposed a scheme that combines a provably secure scheme and identification (ID)-based scheme to provide authentication, anonymity, and non-repudiation. Unlike the original ID-based signature scheme, the proposed scheme generates a member's public key from its identity information (e.g., email address, name, network address, etc.). As an advantage of the scheme, the group controller uses the smaller ID rather than the larger public key, as used by a public key infrastructure-based scheme, to generate a member's private key in order to reduce the storage overhead. A member signs the message with its private key on behalf of the group.

Lee [13] proposed a threshold signature scheme with multiple signing policies. The scheme enables the group signature-generating functionality to be shared among at least any t members out of n members, so that a threshold value of the signature is t. Any t-1 or lower members cannot generate or reconstruct the same signature with the threshold value of t. This proposed scheme demonstrated that each user stores only a

group secret key (called a public shadow), thereby significantly reducing the storage and communication overheads for group signature generation.

Ateniese *et al.* [14] proposed a provably secure group signature scheme and a modified identity escrow scheme. The proposed scheme enables a group member to authenticate the new comers by using the zero knowledge proof method before issuing a membership certificate. In addition, the scheme allows group members to perform group signing by showing the proof of knowledge of their certificates. Using a modified identity escrow scheme, the receiver does not know of the signer's identity but is guaranteed by the third trusted verifier that the signer's signature can be opened and linked to the signer. Thus, a signer does not expose its secret to the verifier during the verification process. Furthermore, the scheme is provably coalition-resistant against an adaptive attacker who can adaptively run the joining process as multiple new members in order to obtain sufficient information to generate valid group certificates.

2.6.4 Group Anonymity

Many articles proposed solutions to provide anonymity in unicast communications, but those solutions may not be suitable to group communications in the following ways: 1) a node has to hide from multiple nodes; 2) group membership management becomes challenging for providing anonymity; 3) a much complicated group key management is needed to anonymously generate, distribute, and manage multiple keys, including the group key and other key-protected keys, so that anonymity in group communications can be possibly pursued. The following schemes attempt to provide anonymity in group communications:

Xiao *et al.* [15] proposed the mutual anonymous multicast (MAM) protocol that allows communications among three types of nodes: anonymous member (AM), non-anonymous member (NM), and middle outsider (MO) nodes. Initially, a set of NM nodes form the anonymous multicast tree. Then, an AM node sets up connections with three possible choices: NM nodes on the tree that can still accommodate connections – called unsaturated NM nodes, unsaturated AM nodes on the tree, and MO nodes that are invited to join the tree. The protocol combines the well-known reverse Onion protocol [16] and Crowd protocol [17] in the following ways. Each AM node creates a remailer as a list of intermediate nodes whose identities are encrypted with their associated public keys in layers, similar to layers of an onion. The NM nodes on the tree keep all remailers associated with a particular AM node. The packet originated from or destined to an AM node will be forwarded through the remailer associated with this AM node. For the AM-to-NM connections, an intermediate node chooses either to deliver the packet directly to the NM node or randomly forward the packet to another node, according to the predefined forwarding probability. For the AM-to-AM connections (mutual anonymous connections), the AM 1 node will select one of its middle nodes to establish a connection with one of AM 2's middle nodes.

Grosch [18] provided both sender and receiver anonymity to multicast traffics through both dedicated and shared anonymisers. The anonymiser receives messages from a sender, processes the messages (for the purposes of integrity, confidentiality, and anonymity), and forwards them as its own messages to receivers via a secured multicast channel. The scheme determines the location of the anonymiser on the multicast tree in such a way that the network loads and average distance (i.e., the average number of links)

2.6.5 Secure Routing

In this service, the IP-based multicast network is mainly considered. A single packet is delivered through multicast routers to a large group of receivers. If an attacker can join a multicast group and launches either passive or active attacks, these attacks would effectively incur high overheads and network-wide failures and unavailability.

Unfortunately, many GCSs assume that the group routing structure (e.g., multicast tree) is secure and unauthorized users neither send nor receive the messages. However, a few secure group routing schemes have been proposed to safeguard the routing infrastructures physically and logically.

Shields and Aceves [20] proposed a keyed hierarchical multicast routing protocol (KHIP) that allows only authorized and trusted members with proper privileges to access and update the multicast tree, thus preventing unauthorized users from joining or expanding the multicast tree. Data messages are protected with random encryption keys and branch keys, but there is no shared group key for the entire multicast group. A member who serves as the core for each branch reprocesses all passing messages as their origin before forwarding them to the parent and children branches of the multicast tree. It was demonstrated that a minimal number of nonces are added to the headers of data and routing updated messages to prevent the replay attack and the forgery attack. Furthermore, the impact of denial-of-service attacks undertaken by untrusted members (e.g., untrusted multicast routers) could be minimized.

Shim [21] introduced a secure multicast routing protocol based on intra-domain and inter-domain routing protocols. The network is divided into domains, each managed by a core router, and all controlling messages associated with the domain are encrypted

from the anonymiser to all receivers can be optimized. To find such an optimal location, the scheme first selects a candidate node in the undirected graph. Then, it assigns the weight of each link on the graph as the number of all receivers that are connected downstream. Based on the link weights, the shortest paths from the candidate node to all receiving nodes are determined, and the multicast tree is formed. To reduce the network load, all nodes can be grouped in a smaller group size, although fewer nodes can then be selected as anonymisers. Given the specified pair of the candidate node and group size value, the scheme calculates the overall weight for this multicast tree as the sum, over all links, of the probability that each link is used. Repeat the process for all combination of candidate nodes and group size values. The node with the lowest overall weight is selected as the anonymiser.

Dolev and Ostrovsky [19] proposed the XOR tree-based scheme to provide efficient anonymous multicast (either sender anonymity or receiver anonymity or both) and to protect the multicast network against the traffic analysis and collusion attacks. The idea is that a forwarding member performs an XOR operation bit-by-bit on data stream forwarded to its predecessor with pseudo-random stream in order to hide the true bits of the data stream. It is analytically demonstrated that the communication overhead on each link and the computational overhead incurred at any member on the forwarding path is greatly reduced.

with a domain control key. All domains are managed by the center router in a hierarchical tree manner. A non-member user is only able to send data messages encrypted with its corresponding sender specific key. All members use the shared group data key to encrypt and decrypt data messages sent by members and the sender specific key (SSK) to decrypt data messages sent by an associated non-member user. The protocol is claimed to achieve scalability and prevent several active and passive attacks, including unauthorized joining the routing multicast tree.

2.7 Group Communications-oriented Networks

This section reviews some SGC frameworks which have been implemented on several existing networks, including multi-agent systems (MASs), personal area networks (PANs), and IP multicast networks.

2.7.1 Multi-Agent System

MAS fundamentally consists of three components: agents, hosts, and controllers/coordinators. An agent can be a software code that runs on a host, operates in an autonomous manner, interacts with other agents, and connects to one or multiple agent coordinators. MAS is agent-based, implying that security services must be provided to end-to-end communications at the agent level, not the host level. Keys and other group-related resource and information are usually stored in agents, and protected from hosts on which these agents operate

Li and Lan [22] proposed a mobile agent system operating through a secure and high performance agent-based multicast network. The proposed solution adopts the concept of multicast by supporting communications between an agent coordinator and an

agent on a host or communications among agents at the agent level. The security solution uses two keys: a group key and a secret key. The centralized agent coordinator generates a secret key that is then cryptographically separated into secret key shadows, each shared individually by an agent. The key management is based on the concept of (k, n) threshold secrecy, by which the secret is shared among n agents and, to reveal the secret, the shares must be obtained from at least k agents. The secret key shadows are used to derive the group key. The group key and secret key shadows are protected from the resided hosts. The coordinator can evict any agent and takes charge of rekeying by excluding the secret key shadow of the evicted agent from the original secret key. The existing agents can correctly compute the new group key while the evicted agent cannot.

Pros: Key secrecies are provided. Security and performance analysis of the proposed solutions are shown. The scheme was claimed to have reduced communications and processing overheads without solid proofs.

Cons: The scheme requires *a priori* embedment of the secret key shadow in the agent, and the scheme makes a weak assumption that an agent is well protected via cryptographic means and that each agent is trusted.

2.7.2 Personal Area Network

PAN communications is enabled by either wired technologies (i.e., USB and Firewire) or wireless technologies (i.e., Bluetooth, Infrared, and Wi-Fi) or a combination of both. In PANs, devices can communicate to each other and form the network around the person. Data can be passed through from one device to others or conveyed to other networks. Data can be encrypted by using a group key that is shared by all devices. The number of nodes is generally not large, and most devices are operated by one person. With such

characteristics, the central authority can be most effective in key management, and membership changes may not be frequent.

Shin *et al.* [23] proposed a framework consisting of key exchanged protocols against a compromised insider device: Leakage-Resilient and Forward-Secure Authenticated Key Exchanges 1 and 2 (LRFS-AKE1 and LRFS-AKE2). The proposed protocols require a centralized server, which exchanges two long-term secret elements with a user: one for authentication (called the verification data), and another for securing its pair-wise communication (called the symmetric key). The group key generation and distribution is executed in three phases. In the first phase, following the LRFS-AKE1, the server and a user verify themselves by using the verification data, i.e., a combination of a random number and the user's password, along with the symmetric key and the list of devices. Subsequently, a pair-wise session key is generated individually for each device. In the second phase, following LRFS-AKE2, each device performs a contributory group key generation in an orderly manner, assisted by the server but without a user interaction. The session key is used to secure the distribution of its contributed key portion. In the third phase, the same group session key is generated independently by each user.

Pros: Threat models are discussed; proposed protocols are suitable for PANs.

Cons: No rekeying mechanism exists; group key secrecy is not fully guaranteed due to the same password being used by the same user; a symmetric key is assumed to be done off line; additional user password is required; a centralized server is ignorantly assumed as trusted; and no membership change protocol exists.

2.7.3 Multicast Security in IP Multicast Networks

Since 2000, the IETF MSEC working group aims to standardize secure group communication protocols over internets with at least three security objectives [24]: first, providing fundamental security services, such as group key management, access control, group authorization, group policy management, and user and message authentication; second, extending operability from centralized networks to distributed networks where multiple trusted entities are deployed throughout networks; third, defending against network-based attacks.

Currently, IETF MSEC WG has published many Request For Comments (RFCs) in three sets of documents, as in [24], as follows: first, a set of fundamental basics of multicast security, such as security requirements and interpretations (RFC 3547) and multicast security architecture (RFC 3740); second, a set of architectures, such as group key management architecture (RFC 4046) and group policy management architecture (first mentioned in RFC 3740); third, a set of protocols that implement each separated solution of the multicast security, such as source authentication (RFC 4082), group key management (RFC 4535), group security policy token (RFC 4534), and multicast extensions to IPSec (RFC 5374).

Chaddoud and Varadharajan [25] proposed a secure source-specific multicast communications (S-SSM) architecture that offers two security services: access control and data integrity for commercial content delivery. It operates by using the protocol independent multicast-SSM (PIM-SSM) routers which form the backbone. S-SSM divides the whole service area into domains and has two layers of controls: the domain-wide level via local controllers, and the network-wide level via a global controller. The

global controller and content distribution server are connected directly to the PIM-SSM routers. The global controller manages data distribution, authorizes user access, generates channel keys, and authorizes rekeying. To manage subscribers, the Internet group management protocol (IGMP) version 3 is deployed in some PIM-SSM routers, which are located outside the backbone and connect directly to subscribers. In the IGMPv3/PIM-SMM routers, the local controller functionality is added to authenticate new subscribers, to distribute a channel key to subscribers, and to periodically rekey the channel key as authorized by the group controller.

Pros: Computation overhead is roughly analyzed based on the number of computing operations; group key management, access control, traffic confidentiality and integrity, and authentication services are offered; and dynamic membership is supported.

Cons: There is insufficient information about communication and computation overheads, and security analysis to substantiate the claim that the proposed scheme is very inefficient; and some communications on the off-line channel may be required.

2.8 Challenging Factors in Designing Secure GCSs

As illustrated in Tables 2.2 and 2.3, there is no unique scheme or system that can achieve all security requirements. Here, various perspectives and attributes in designing a secure and high performance GCS will be summarized shortly.

2.8.1 Environment and System Performance

From the perspective of group management, a central group controller (and, in some systems, a key server) in a centralized GCS can afford intensive computations and storage overhead but, in return, becomes the point of attack which threatens to shutdown

all group operations. The other upsides are that high security can be achieved effectively and quickly, and each group member sustains less workload. A decentralized GCS reduces the workload performed by each sub-group controller. The apparent downsides include an additional communication overhead caused by communications among sub-group controllers, and the single point of failure problem. A distributed scheme increases workloads for each member in terms of storage and computation overheads, although the system is more scalable and eliminates the need of the central authority. For either environment (centralized or distributed), the design should optimize the system performance measured in terms of overheads (communication, computation, and storage) burdened on each group member, the key server, and the controller of the system.

2.8.2 Efficiency of Key Management and Distribution

The efficiency of several security services relies on the strength of the key management and the cryptographic strength of the keys. An efficient GKM scheme should mainly reduce time complexity and computation load of key generation, key distribution, and rekeying. The scheme should be scalable as the group size increases. Many efficient GKMs generate keys based on the structure of a key tree and the hierarchy of keys, especially for centralized and decentralized environments. In a distributed environment, a contributory GKM scheme deems to be more suitable.

2.8.3 Early Detection and Prevention

The secure GCS should be operated with strong authentication and access control mechanisms by which a violation of resource utilization and unauthorized activities, e.g., a member impersonation and a message fabrication, can be detected early and prevented.

A group signature signed on messages can also provide source authentication, message integrity, and non-repudiation services to the receiver and verifier. Since communication, storage, and processing overheads are the primary cost for these security services, a trade-off between overheads and the protection level should be properly optimized.

2.8.4 Increased Concern over Privacy

Privacy becomes a major concern for users participating in group communications where there are a large number of message recipients so that message confidentiality may not be fully guaranteed, and security enforcement may not be possible or adequate. In general, anonymity service substantially increases overheads and the complexity. Thus, it may not be suitably in deployed in distributed environments where resources are scarce. Instead, partial anonymity can be utilized in such a way that, for a large group, only partial identification is required to prove a rightful communication while it still preserves member privacy.

2.8.5 Implementation of Security Services for Different Applications

From the perspective of group-oriented applications, security services should be offered and compatibly interacted for any applications to achieve a high security level. Thus, the system should be transparent to applications.

2.9 Conclusion

Group communications has received considerable attention owing to its nature to transmit a single message to multiple recipients with the minimum bandwidth overhead. The objective of this chapter is to provide a better understanding on possible attacks, security requirements, and security services in group communications. Figures 2.2 and 2.3

identify fundamental attributes for evaluating mechanisms in providing one or more security services to GCSs as well as attributes corresponding to those supported security services. Then, based on attributes in Figures 2.2 and 2.3, the comparisons of these outstanding secure GCSs are summarized in Tables 2.2 and 2.3. There are some challenges for designing secure group communications, such as system-wide performance, efficiency of key management, privacy issue, implementation, and tradeoff between security and overheads. Readers are further referred to Reference [2] for a survey on secure group communications in wireless networks, such as mobile ad hoc networks and wireless sensor networks, for a broader understanding on secure group communications.

Table 2.2 The Comparison Table of Secure GCSs Along With Fundamental Performance-evaluating Attributes [1]

Services / Evaluating properties		Group Key Management				Group Access Control		Group Signature			Anonymity		Secure Multicast Routing	
		Ref. 2	Ref. 3	Ref. 4	Ref. 5	Ref. 6	Ref. 7	Ref. 8	Ref. 9	Ref. 10	Ref. 11	Ref. 12	Ref. 13	Ref. 14
Type of group management		Centralized (a group manager) + distributed (key holders)	Partially distributed (an authentication and access control server + cluster leaders)	Distributed (subgroups)	Distributed (logical servers)	Centralized (with a key distribution center) + contributory scenarios	Any (with a key server)	Centralized (a group manager)	Centralized (with a shadow distribution centre)	Centralized (a group manager)	Centralized (a group authority and a group controller)	Fully distributed	Partially distributed (sub-branch core routers, and an authentication service)	Partially distributed (domain core routers + a center router)
Objective for a reduction of storage overheads		Yes	Yes	Yes	No	Yes	Yes	No	Only one secret kept at each member	No	Yes	No	No	No
Objective for a reduction of communication overheads		Yes	Yes	Yes	Yes	N/A	N/A	No	No	No	Yes	Yes	No	No
Objective for a reduction of processing overheads		Yes	Yes	Yes	Yes	Yes (for rekeying)	Yes (for rekeying)	No	No	No	Yes	Yes	No	No
Performance is steady with a group size		Yes	Yes	Yes	Yes	Yes	Possible but not clearly stated	N/A	N/A	N/A	No	Yes	No	No (for edge/ core/ center routers)
Scalability supported		Yes	Yes	Yes	Yes	Yes	Possible but not clearly stated	No	No	No	Yes	Unlikely but not clearly stated	Yes (for key generation and management) No (for rekeying)	Yes
Dynamic membership	Individual change	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	N/A	N/A	Yes	Yes
	A bulk of changes	Yes	Yes	No	Yes	No	No	No	No	No	No	No	No	No
Trust among group entities required		No	Yes (among members)	Yes (for key server(s))	No	No	No	No	No	Yes (for the verifier(s))	No	No	Yes	No
Message integrity		Traffic encryption key and key encryption keys	N/A	Session key	Group key	Session key and key-encrypted keys	Data encryption keys (resource and membership-group keys)	N/A	N/A	N/A	Message encryption and session keys	Pseudorandom sequences	Random encryption keys + branch keys	Hash of encrypted messages + domain control key + group data key + sender specific keys)
Attacks	Possibly vulnerable to following attacks	Single point of failure and setup implosion [centralized] + collusion and insider attacks [distributed]	Single point of failure (at the authentication and access control server), replay, collusion, and denial-of-service attacks	Single point of failure (at the key server(s)) . replay, collusion, and denial-of-service attacks	Collusion, byzantine, insider, and denial-of-service attacks	Single point of failure (at the key distribution center), identity, replay, and denial-of-service attacks	Single point of failure (at the key server), impersonation, identity, replay, and denial-of-service attacks	Single point of failure (at the group manager), replay, and denial-of-service attacks	Single point of failure (at the shadow distribution center), replay, and denial-of-service attacks	Replay and denial-of-service attacks	Single point of failure, collusion, denial of service, cut-and-paste, insider, and byzantine attacks	Insider, Byzantine, and identity attacks	Single point of failure, denial-of-service, and eavesdropping attacks	Single point of failure (at the center router) and denial-of-service attacks
	Assuredly resilient to attacks	Node failures, replays, and unauthorized break-ins attacks	Replay and eavesdropping attacks	Impersonate and message fabrication attacks	Single point of failure attack, node and network failures	Unauthorized break-ins	Unauthorized break-ins	Collusion, framing, and message fabrication attacks	Impersonate, framing, and message fabrication attacks	Collusion, framing, and impersonate attacks	Traffic analysis and identity attacks	Eavesdropping , collusion, and back-off attacks	Replay, fabrication, cut-and-paste, denial-of-service, and flooding attacks	Flooding, replay, insider, and message fabrication attacks

Table 2.3 The Comparison Table of Secure GCSs Along With Service-specified Performance-evaluating Attributes [1]

Evaluating properties	Group Key Management				Group Access Control		Group Signature			Anonymity		Secure Multicast Routing	
	Ref. 2	Ref. 3	Ref. 4	Ref. 5	Ref. 6	Ref. 7	Ref. 8	Ref. 9	Ref. 10	Ref. 11	Ref. 12	Ref. 13	Ref. 14
KEY MANAGEMENT													
Structure of keys	Tree based & Table based	Hierarchical cluster-based	Key graph	Typical group key agreement	Tree-based Multi-group key graph	Logical key graph						Hierarchical branch-based tree	Hierarchical domain-based tree
Type of cryptosystem used	DL	DL	RSA	DL and RSA	N/A	N/A						N/A	RSA
Forward secrecy	Yes	Yes	Yes	Yes	Yes	Yes						Yes	Yes
Backward secrecy	Yes	Yes	Yes	Yes	Yes	Yes						Yes	Yes
Perfect forward secrecy	N/A	N/A	N/A	Yes	N/A	N/A						N/A	N/A
Key serialization	No	No	No	No	No	No						No	No
Key independence	Yes	Yes	Yes	Yes	N/A	N/A						N/A	N/A
Rekeying	Membership changed	Membership changed	Membership changed	Membership changed	Membership and access privilege changed, and periodic	Membership changed						Membership changed and periodically changed	Membership changed
Key management	Centralized (a group manager) + distributed (temporary key holders)	Centralized (a key server)	Centralized (a key server)	Contributory	Centralized (a key distribution center)	Centralized (a key server)						Partially distributed (sub-branch routers)	Partially distributed (domain routers)
AUTHENTICATION													
User authentication	Yes	Yes (auth. list + credential.)	Yes	No			Yes (Membership cert. + zero-knowledge proof)	No	Yes (Membership cert. + zero-knowledge proof)	Possible but not clearly stated	No	Yes (Cert.)	Yes
Message authentication	Yes	No	Yes	Yes			Yes (group signing keys)	Yes (group secret keys)	N/A	Yes	No	Yes	Yes
ACCESS CONTROL													
Authorization/ access control	Yes	Yes	Yes (access control list)	No	Yes (hierarchical access control)	Yes (hierarchical access control)				Yes (access control cert. and list + access control anonymiser)	No	Yes (access control list.)	Yes (an authorization service + access control list)
Dynamic access control	No	No	No	No	Yes	No				No	No	Yes	No
SIGNATURE													
Message signature	No	No	Message digest	Message signature			ID-based group signature	Threshold-based group signature	Group signature	Messages signature	No	Digital signature	No
Non-repudiation	N/A	N/A	Unforgeable and linkable	Unforgeable and linkable			Unforgeable, non-allegeable, and linkable	Unforgeable, non-allegeable, and linkable	Unforgeable, non-allegeable, and linkable	N/A	N/A	N/A	N/A
ANONYMITY													
Anonymity supported							Yes	Yes	Yes	Yes	Yes	No	No
Unlinkability of anonymous communications							Sender	Sender	Sender	Sender-receiver	Sender, receiver, and sender-receiver		
Type of anonymity management							N/A	N/A	N/A	Centralized (anonymisers)	Distributed		
SECURE ROUTING													
Management												Partially distributed (sub-branches)	Partially distributed (domains)
Prevention												Unauthorized modification	Unauthorized modification

CHAPTER 3

SECURITY SERVICES IN GROUP COMMUNICATIONS OVER WIRELESS INFRASTRUCTURE, MOBILE AD-HOC, AND WIRELESS SENSOR NETWORKS

3.1 Objective

Group communications in wireless networks has been facilitating many emerging applications which require packet delivery from one or more sender(s) to multiple receivers. Owing to insecure wireless channels, group communications are susceptible to various kinds of attacks. Though a number of proposals have been reported to secure group communications, provisioning security in group communications in wireless networks remains a critical and challenging issue. This chapter presents a survey on recent advances in security requirements and services in group communications in three types of wireless networks, and discusses challenges in designing secure group communications in these networks: wireless infrastructure networks, mobile ad hoc networks, and wireless sensor networks.

3.2 Introduction

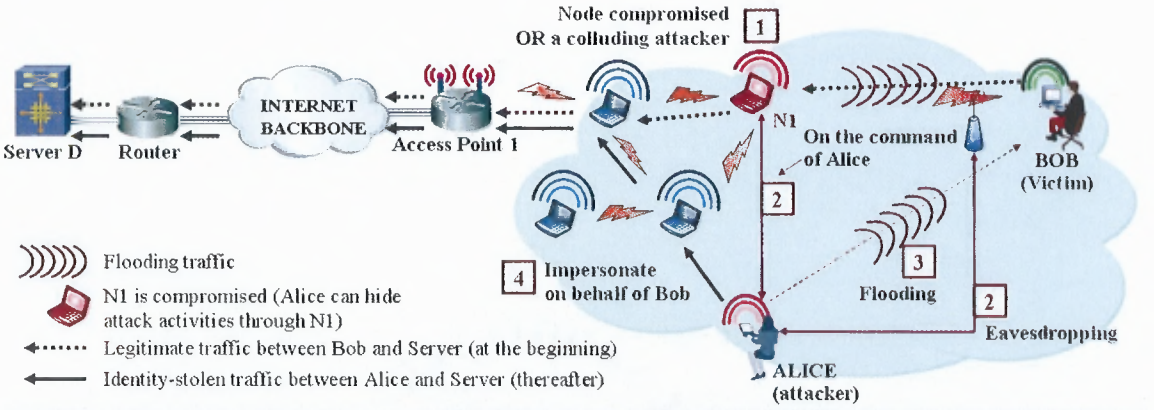
Group communications refers to either point-to-multipoint (in which a packet is delivered from a group member to the other members) or multipoint-to-multipoint communications (in which packets are sent from multiple members to other members simultaneously). The characteristics of different wireless networks, i.e., wireless infrastructure networks (WINs), ad hoc networks (AHNs), and wireless sensor networks (WSNs), are vastly different in terms of group management, packet types, and resources. However, one common risk among these networks is that all members communicating through wireless

channels are more insecure and susceptible to numerous attacks, as compared to wired networks [26]-[28]. Thus, an attempt to establish secure group communications (SGC) over these networks faces various challenges in order to meet security requirements as specified in Table 3.1.

The rest of this chapter is organized as follows. First, various known attacks are presented, followed by a discussion on group communications, and security requirements and services in securing group communications. Several proposals for SGC over these different networks are then discussed.

3.3 Known Attacks in Wireless Networks

Here, this work presents some known attacks (intensively discussed in [27]-[30]), and sparsely discussed in other references) that pose a significant threat to group communications over wireless networks, and categorize these attacks based on their impacts, including data integrity and confidentiality, power consumption, routing, identity, privacy, and service availability. For example, Figures 3.1 and 3.2 illustrate some of these attacks in a real wireless network.



Scenario: Bob communicates with a data-sensitive server through hybrid (infrastructure/ad hoc) wireless networks.
Attacks: Eavesdropping, impersonate, flooding, and collusion.
Steps:
1A. Alice has previously compromised N1 by any means.
1B. Alternatively, N1 is a colluding attacker that shares information and helps to carry out the attack.
2. Alice may command N1 to start eavesdropping on Bob's incoming and outgoing packets. If possibly, Alice may eavesdrop on N1 packets due to insecure wireless channel. This eavesdropping attack aims to collect more information about Bob's communication with the Server D for traffic analysis.
3. Alice floods Bob's machine or network such that Bob is denied of service. Note that Alice is unlikely to ask a colluding N1 to launch the flooding attack because N1 will possibly be caught.
4. Alice impersonate Bob's identity to falsely gain access, on behalf of Bob, to information authorized by server D to Bob . Note that Bob cannot notify server D due to the effect of the flooding attack and we assume that Bob does not use an out-of-bound channel (e.g., telephone to the server D's operator).

Figure 3.1 An illustration of mixed attacks in a real wireless network [2].

3.3.1 Data Integrity and Confidentiality-related Attacks

In general, this type of attacks attempts to reveal or compromise the integrity and confidentiality of data contained in the transmitted packets.

A. Denial of service on sensing (DoSS) attack – an attacker tampers with data before being read by sensor nodes, thereby resulting in false readings and eventually leading to a wrong decision. This DoSS attack generally targets the physical layer applications in an environment where sensor nodes are located.

B. Node capture attack – an attacker physically captures sensor nodes and compromises them such that sensor readings sensed by compromised nodes are inaccurate or manipulated. In addition, the attacker may attempt to extract essential

cryptographic keys (e.g., a group key) from wireless nodes which are used to protect communications in most wireless networks.

C. Eavesdropping attack – an attacker secretly eavesdrops ongoing communications between targeted nodes to collect information on connections (e.g., MAC address) and cryptography (e.g., session key materials). Although this attack can be classified into other categories such as privacy-related attacks, it is grouped into this category owing to its severe consequence in the sense that the collected cryptographic information may break the encryption keys such that the attacker can retrieve meaningful data.

3.3.2 Power Consumption-related Attacks

In general, this type of attack attempts to exhaust the device's power supply, which is one of the most valuable assets in wireless networks. The worst case would cause a collapse of network communications.

A. Denial of sleep attack – an attacker tries to drain a wireless device's limited power supply (especially sensor devices) so that the node's lifetime is significantly shortened. In general, during a sleep period in which there is no radio transmission, the MAC layer protocol reduces the node's power consumption by regulating the node's radio communications. Thus, the attacker attacks the MAC layer protocol to shorten or disable the sleep period. If the number of power-drained nodes is large enough, the whole sensor network can be severely disrupted.

3.3.3 Service Availability and Bandwidth Consumption-related Attacks

These attacks can, in fact, also be categorized into power consumption-related attacks. However, since they mainly aim to overwhelm the forwarding capability of forwarding nodes or consume sparsely available bandwidth, they are more likely related to the service availability and bandwidth consumption concerns, and are thus highlighted in this category. If these attacks result in a denial of service to legitimate members, they can also be referred to as a variant of denial-of-service (DoS) attacks.

A. Flooding attack – an attacker typically sends a large number of packets to the access point or a victim to prevent the victim or the whole network from establishing or continuing communications.

B. Jamming (radio interference) attack – an attacker can effectively cut off wireless connectivity among nodes by transmitting continuous radio signals such that other authorized users are denied from accessing a particular frequency channel. The attacker can also transmit jamming radio signals to intentionally collide with legitimate signals originated by target nodes.

C. Replay attack – an attacker copies a forwarded packet and later sends out the copies repeatedly and continuously to the victim in order to exhaust the victim's buffers or power supplies, or to base stations and access points in order to degrade the network performance. In addition, the replayed packets can crash the poorly designed applications or exploit the vulnerable holes in poor system designs.

D. Selective forwarding attack – a forwarding node selectively drops packets that have been originated or forwarded by certain nodes, and forwards other irrelevant packets instead.

3.3.4 Routing-related Attacks

In general, these attacks attempt to change routing information, and to manipulate and benefit from such a change in various ways.

A. Unauthorized routing update attack – an attacker attempts to update routing information maintained by routing hosts, such as base stations, access points, or data aggregation nodes, to exploit the routing protocols, to fabricate the routing update messages, and to falsely update the routing table. This attack can lead to several incidents, including: some nodes are isolated from base stations; a network is partitioned; messages are routed in loop and dropped after the Time-To-Live (TTL) expires; messages are perversely forwarded to unauthorized attackers; a black-hole router in which messages are maliciously discarded is created; and a previous key is still being used by current members because the rekeying messages destined to members are misrouted or delayed by false routings.

B. Wormhole attack – an adversary intercepts communications originated by the sender, copies a portion or the whole packet, and speeds up sending the copied packet through a specialized "wormhole tunnel" such that the copied packet arrives at the destination earlier than the original packet traversed through normal routes. The wormhole tunnel can be created by several means, such as by sending the copied packet through a wired network and at the end of the tunnel transmitting over a wireless channel, by using a boosting long-distance antenna, by sending through a low-latency route, or by using any out-of-bound channel as illustrated in Figure 3.2. The wormhole attack poses many threats especially to routing protocols and other protocols that greatly rely on geographic location and proximity, and

many subsequent attacks (e.g., selectively forwarding, sinkhole, etc.) can be launched after the wormhole path has attracted a large number of traversing packets. Readers are referred to Reference [29] for details and a mechanism to detect such an attack.

C. Sinkhole attack – an attacker attracts all nodes to send all packets through one or several of its colluding nodes, called sinkhole node(s), so that the attacker (and its colluding group) has access to all traversing packets. To attract the victimized nodes, the sinkhole node is usually presented as an attractive forwarding node such as having a higher trust level, being advertised as a node in the shortest-distance or short-delay path to a base station, or a nearest data-aggregating node (in WSNs).

3.3.5 Identity-related Attacks

In general, these attacks cooperate with eavesdropping attacks or other network-sniffing software to obtain the vulnerable MAC and network addresses. They target the authentication entity.

A. Impersonate attack – an attacker impersonates another node's identity (either MAC or IP address) to establish a connection with or to launch other attacks onto a victim; the attacker may also use the victim's identity to establish a connection with other nodes, or to launch other attacks on behalf of the victim as illustrated in Figure 3.1. There are several softwares capable of reprogramming the devices to forge the MAC and network addresses.

B. Sybil attack – a single node presents itself to other nodes with multiple spoofed identifications (either MAC address or network address). The attacker can impersonate other nodes' identities or simply create multiple arbitrary identities in

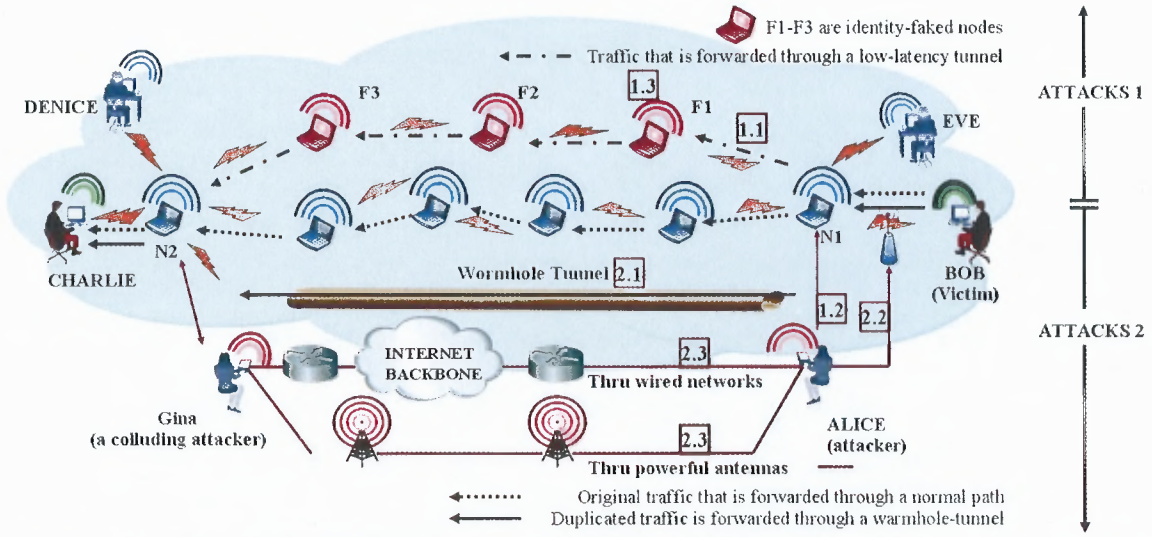
the MAC and/or network layer. Then, the attack poses threats to other protocol layers; for examples, packets traversed on the route consisting of fake identities are selectively dropped or modified; or a threshold-based signature mechanism that relies on a specified number of nodes is corrupted.

3.3.6 Privacy-related Attacks

In general, this type of attack uncovers the anonymity and privacy of communications and, in the worst case, can cause false accusations to an innocent victim.

A. Traffic analysis attack – an attacker attempts to gain knowledge about the network, traffic, and nodes' behaviors. The traffic analysis may include examining the message length, the message pattern or coding, and the duration of the message stayed in the router. In addition, the attacker can correlate all incoming and outgoing packets at any router or member. Such an attack violates the privacy and can harm the members for being linked with messages (e.g., religious-related opinions that are deemed provocative in some communities). The attacker can also perversely link any two members with any unrelated connections.

If a group of attackers collude to launch any type of attacks, it is referred to as a collusion attack. For examples, the colluding group of attackers orchestrates to collect information to significantly exploit the system, to masquerade a legitimate member and send out fault messages on behalf of that member, to conjointly mount attacks against other members or network entities, or to falsely accuse a legitimate member as an attacker.



Scenario: Bob (and Denice) communicates with Charlie (and Eve) through a mobile ad hoc network.

Attacks 1: Sybil, sinkhole, and selectively forwarding attacks.

Steps:

1.1. Alice generates multiple fake identifications, and advertises an attractive "low-latency" path consisting of these identities to N2. (Sybil attack)

1.2. Alice advertises this low-latency path to N1, who updates its routing table accordingly. N1 forwards Bob's packets to Charlie through this low-latency path. (Sinkhole attack)

1.3. Alice passively eavesdrops all Bob-Charlie's packets or selectively drops all Bob-Charlie's packets while still forwarding Denice-Eve's packets to reduce the possibility of being caught. (Selectively forwarding attack)

Attacks 2: Eavesdropping, wormhole, false routing update, and collusion attacks.

Steps:

2.1. Alice sets up a wormhole tunnel (in this illustration, either thru wired networks or thru powerful antennas) to Gina, a colluding attacker. (Collusion attack)

2.2. Alice eavesdrops Bob-N1's packets and copies them. (Eavesdropping attack)

2.3. Alice forwards the copies to Gina through the wormhole tunnel. (wormhole attack)

2.4. Gina forwards the copies to N2. N2 updates the new route between N1 and N2 through this wormhole tunnel (false routing update).

Figure 3.2 An illustration of the routing-related attacks and other attacks [2].

3.4 Secure Group Communication Systems

A group communication system (GCS) consists of five common operations: initiate, join, leave, partition, and merge. The group is first established by initial members. Then, one or several prospective members join the group while some members leave the group. This is so-called the dynamic membership. A large number of membership changes, referred to as a bulk membership change, require a specialized protocol design without degrading the group performance. In some scenarios, a group can be partitioned into smaller subgroups or merged into a bigger group. This can also be considered as a bulk membership change, but the transitions among groups likely incur overheads. This

dynamic membership aspect requires the GCS to rekey the session keys in order to preserve the key secrecy. For WSNs, this dynamic membership may not be necessary because the keys are most likely pre-determined prior to deployment [31]-[33]. In wireless infrastructure and ad hoc networks, most, if not all, GKM schemes require each member to keep the membership list, thus incurring a huge communication overhead. However, in a wireless sensor network, this list might not be necessary due to 1) storage limitation of sensor nodes, and 2) the provision of pre-selected entities (data-aggregating nodes) in keeping track of their members.

Table 3.1 Characteristics of Possible Attacks on SGC over Wireless Networks [2]

Networks / Characteristics	Wireless infrastructure networks	Mobile ad hoc networks	Wireless sensor networks
Central authority	Yes	No	Yes (base stations/data aggregation nodes)
Computation	High / Varying (base stations / devices)	Varying	High / Very low (base stations / sensors)
Storage	High / Varying (base stations / devices)	Varying	High / Very low (base stations / sensors)
Power Supply	High	Varying	Low
Handoff	Yes	No (in IPv4)	Not likely
Mobility (dynamic membership)	Varying	High	Varying (Likely fixed)
Network topology	Varying (Likely low dynamic)	Highly dynamic	Varying (Likely highly dynamic due to short life-cycles and unreliability of nodes)
Message length	Varying (depending on applications)	Varying (depending on applications)	Relatively short and aggregated
Connectivity	Continuous	Likely short live	either shortly periodic or continuous
Direction of connections between a member and a designated controller	(a member – an access point) Duplex	(a member – a designated controller) Duplex	(a sensor node – an aggregator) Uniplex for most communications Duplex only in certain incidents (e.g., locating the aggregator)
Key management	Centralized / contributory	Distributed / contributory	Distributed / contributory
Pre-determined information	Possible	Limited – not at all	Most likely
Potential cryptography	Varying	Varying	Symmetric / Elliptic Curve cryptography (ECC)
Additional criteria	1. Key management and dynamic membership during a handoff period.	1. All network and key management tasks should be equally distributed (fairness). 2. Network partitions can occur more frequently and may increase the number of isolated nodes. 3. Multicast routings are updated frequently due to a dynamic network. 4. Trust relationship can enhance the performance.	1. The ratio of the length of encrypted messages-to-messages should be as low as possible. 2. Trust relationship can enhance the performance.
Some known attacks	Single point of failure (at access points), Denial of Service, Collusion, Insider, Traffic analysis, Routing, Identity, Replay, Jamming.	Denial of Service, Collusion, Insider, Traffic analysis, Routing, Identity, Replay, Jamming, Sybil, Wormhole, Sinkhole.	Single point of failure (at base stations/ data-aggregating nodes), Denial of Service, Collusion, Insider, Traffic analysis, Routing, Identity, Replay, Jamming, Sybil, Sinkhole, Denial of sleep, Denial of service on sensing, Node capture.

3.5 Security Requirements and Security Services in SGC

This section discusses security requirements and corresponding security services in securing group communications and mitigating attacks as summarized in Tables 3.1 and 3.2. Table 3.3 describes in-depth details of the characteristics of major security services over wireless networks. Many systems have been proposed to address the requirements and provide such services, but only a few promising systems are presented here.

Group Key Management (GKM) – The fundamental security service in SGC is the provision of a shared group key, which is used to encrypt a group message, to sign the message, to authenticate members and messages, and to authorize an access to traffic and group resources. Thus, the strength of SGC largely relies on the cryptographic strength of the keys and the key management protocol. A GKM scheme deployed in any secure group communication system should satisfy the following requirements: key generation is secure; imitation of the group key should be infeasible or computationally difficult to succeed; the group key is securely distributed and only the legitimate users can receive a valid group key; revocation of the group key upon every membership change should be immediate; every membership change must result in rekeying of associated keys; and a rekeying of the key is secure. Basically, GKM can be categorized into the three types based on how the key is generated: centralized, distributed, and contributory [34]-[35]. A revocation can be performed by limiting the session period that the keys are valid. Then, the session period and remaining period are calculated and attached along with keys before being distributed to all members. If keys need to be rekeyed (with three triggering conditions as discussed in Table 3.3), the key revocation can be sent by the designated entity to notify all members holding these keys, as discussed in [31] and [34].

Group authentication – In group communications (1-to-Many and Many-to-Many), a member can be either the designated sender or the designated receiver, or both. Both users and messages should be authenticated to safeguard identity-related attacks. In some systems, a member certificate is issued by the trusted certificate issuing entity along with its validation period. In some systems, the expired certificate is maintained for further verifications, as discussed in [34]. The expired certificates are compiled into the revocation list, which is distributed to notify all members.

Group authorization and access control – In any conventional access control mechanism, a member who holds a decrypting key can access full contents in a flow (or all flows in an aggregated stream). This is referred to as a single access privilege. In many group-oriented applications, group members can be assigned with multiple access privileges. Thus, the stream should be accessed with different access privileges such that only members who have an appropriate privilege can access the corresponding portions of contents (or flows). This is referred to as multiple access privilege.

Group accounting and non-repudiation – Any group operation executed or a record of resources utilized by a member should be available for tracking in order to detect any abusing usage of resources and operations. A non-repudiation service can ensure that the identity of a member whose activities are in dispute can be fully and precisely determined by the designated entity. In general, the group signature and the member certificate can be used to authenticate the source and message, and to provide a proof of the source's activity in the case of a dispute.

Group privacy and Anonymity – Any information related to a group message, such as identities of a sender and a receiver, message length, and time, can be protected

or hidden to preserve privacy and anonymity of members. An anonymous message refers to a message that carries no information about the senders and receivers.

Group message integrity and confidentiality – Message integrity should be preserved by ensuring that the message has not been fabricated (some or all portions of the message have not been added, deleted or modified) or dropped by an unauthorized entity. This can be done by several means, including hashing and signing the message along with strong encryption keys. In ad hoc networks, group members may have different capabilities and protocols to perform different levels of encryption on group messages. Thus, some messages may be encrypted with a strong encryption while others with a weak encryption are relatively easily breakable. In WSNs, sensor nodes may have similar capabilities and protocols that are embedded before deployment. Confidentiality ensures that only authorized members can retrieve meaningful data from the message.

Group Survivability and Availability – An attacker can attack the routing hosts (i.e., access points and base stations) to isolate some or all group members, or partition the group. Thus, all routing hosts must be protected to ensure group survivability. However, the attacker can still target a joining procedure (i.e., by flooding the access point or base station in wireless infrastructure networks and WSNs), thus causing service unavailability to other legitimate users. Group availability ensures that only authorized users can always communicate within the group by using restricted group resources, and any violation of exceeding the limitation of group resources will be promptly detected.

Table 3.2 illustrates each discussed attack along with security services that can be deployed to mitigate its impact. For examples, the impact caused by the flooding attack may be partially mitigated by authenticating sources that generate the flooding packets

along with an early detection of a massive amount of packets originated by a single source. Thus, the flooding packets would be dropped immediately once such an attack has been detected. Unauthorized routing update can be detected and prevented by the following services: authenticating both the source and message to determine whether the routing update message is legitimate and originated by an authorized member; enforcing an access control over routing table; signing the routing update message such that message integrity is preserved and no attacker has falsely modified the message; encrypting all management packets (routing update requests and replies); and any loophole or sinkhole routing, which possibly leads to a denial of service, will be tested, detected, and fixed prior to the actual deployment.

Table 3.2 Security Services to Countermeasure Attacks [2]

Attacks	Security services to countermeasure attacks					
	Authentication	Authorization/ Access control	Accounting/ Non repudiation	Message Confidentiality and Integrity	Privacy/ Anonymity	Survivability/ Availability
Denial of Service on Sensing	--	--	--	--	--	sensing tampering detection
Node Capture	--	username, password, ID	--	e{management & data} & hash	--	node intrusion detection
Eavesdropping	--	--	--	e{management & data} & hash	source - destination anonymity	--
Denial of Sleep	source & message authentication	access control on routing table	group signature	e{management} & hash	--	--
Flooding	source authentication	--	--	--	--	early detection for an excessive amount of packets
Jamming	--	--	--	e{data} & hash	--	jamming detection
Replay	--	--	group signature, timestamp, and packet sequence number	e{data with nonce} & hash	--	--
Selective Forwarding	source & message authentication	--	group signature, timestamp, and packet sequence number	e{data with nonce} & hash	--	--
Unauthorized routing update	source & message authentication	access control on routing table	message signature	e{management} & hash	--	loophole and sinkhole routing detection
Wormhole	source authentication	access control on routing table and using directional antenna	--	e{management, data with nonce} & hash	--	--
Sinkhole	source & message authentication	access control on routing table	--	e{management} & hash	--	--
Impersonate	source authentication	access control list	group signature and time-expired certificate	e{management & data} & hash	--	--
Sybil	source authentication	access control list	group signature and time-expired certificate	e{management & data} & hash	--	multiple IDs detection
Traffic Analysis	message authentication	--	group signature, Timestamp, and packet sequence number	e{data} & hash	source - destination anonymity	--

Note: e{…} = an encryption of …

3.6 SGC over Wireless Infrastructure Networks

This section surveys some SGCs which provide security protection over wireless infrastructure networks.

DeCleene *et al.* [30] presented a hierarchy-based key management protocol that divides an operational field into administratively independent areas. The area key is used

to encrypt the message containing the data key. The data key is a network-wide shared key, and is used to encrypt the data message. When users frequently move within areas, the area key is rekeyed, thereby resulting in a significant degradation of group performance in terms of processing and communication overheads. Thus, several rekeying algorithms have been proposed to reduce the need of rekeying, thus decreasing communication and processing overheads. The delayed rekeying algorithm uses the "extra key owner list (EKOL)" to store the area keys belonged to the leaving member and that member's ID. When that member re-enters the area, the area keys are restored. However, once the EKOL is full, the first recorded area keys are discarded to make room for other members. A member can only hold a limited number of area keys.

Pros: Low overheads; highly dynamic membership is supported.

Cons: The area keys may be easily compromised since the area keys have been repeatedly re-used for often-moving members.

Sun *et al.* [36] matched the tree-based key management with the physical cellular network topology in order to build a topology-matching key management (TMKM) tree. When the user moves among cells, an efficient handoff mechanism handles the relocation of that user in the TMKM tree. Each cell has a corresponding wait-to-be-removed (WTB) list that keeps track of previous and current cell members. A relocation of a member between two cells requires a rekeying process to preserve the key secrecy. The rekeying process is performed based on information from the WTB lists of these cells. The key manager, called key distribution center (KDC), maintains and updates the WTB lists of all cells in the network accordingly. The communication overheads incurred by the rekeying process can be reduced by delivering new keys locally in the TMKM tree to

only members who need the rekeyed keys. It was shown that communication overheads due to the efficient handoff rekeying processes using the TMKM tree scheme can be greatly reduced as much as 80% as compared to those using conventional topology independent key management (TIKM) tree schemes.

Pros: Low communication overheads.

Cons: The scheme does not consider the overheads incurred by the KDC that could result in very poor performance in the actual deployment.

Gupta and Cherukuri [26] presented three schemes: single session key (SSK), different session key (DSK), and a combination (HYBRID). These schemes are based on location-based access control in which only users who are located in specific locations can access the services. In the SSK scheme, a base station (BS) assigns the same session key (*sk*) to all members. In the DSK scheme, a BS assigns a different session key to each member. In the HYBRID scheme, a BS assigns the same *sk* to all the members who have been a "stable" member in the cell for more than a specified period of time; otherwise, it assigns a different *sk* to a "non-stable" member.

Pros: Their simulations of SGC over all cellular networks with high mobility showed that the communication overhead using the HYBRID scheme is lower than those using the DSK and SSK schemes.

Cons: Strict time synchronization is required to determine whether a member is classified as stable or non-stable; the scheme did not provide a means for base stations to exchange information of their members; and handoffs, which can incur more overheads, were not addressed.

Westerhoff *et al.* [38] presented a decentralized architecture called mobility support – a multicast-based approach (MOMBASA) to achieve low latency for handoffs with minimum packet loss as well as to secure protocol operations. MOMBASA enables each mobile node to register with a proxy, called Mobility-Enabling Proxy (MEP), which in turn participates in the multicast group on behalf of the mobile node. The mobile node communicates with MEP via unicast while MEP communicates with the multicast group via multicast. Thus, MEP converts unicast and multicast packets between the mobile node and the multicast group. The security analysis shows that MOMBASA is protected against various attacks by using three security components: packet filtering at access network boundaries, deployment of an authentication, authorization, access control (AAA) infrastructure, and rate limiting against DoS attacks.

Pros: MOMBASA is provably secure from many threats; performance degradation due to handoffs is negligible (low-latency handoff); less packet loss; and the workloads among access points and the AAA server are fairly distributed.

Cons: The scheme only considered handoffs when MEP is no longer functioning, but did not consider the case when the membership is highly dynamic. When there are messages destined for idle nodes, the MEP associated with these nodes has to multicast the paging update messages to other MEPs, thus incurring a significant overhead.

3.7 SGC over Mobile Ad Hoc Networks (MANETS)

This section surveys some SGCs which provide security protection specifically over mobile ad hoc networks.

Kaya *et al.* [39] proposed a dynamic multicast group management protocol that attempts to equally distribute the workload of securing communications to all participating members. Group information and associated security services are disseminated and maintained by all members throughout MANETs, and the service right certificates are given by the designated group manager to members for accessing information. The group manager is temporarily selected per session, and it establishes a physical security tree, authenticates the prospective members, updates the security tree per membership change, and handles the revocation of certificates. The security tree is used to securely forward the shared group key to members while the data multicast tree is used to forward the encrypted data messages to authorized members.

Pros: Communication overheads and latency of joining/leaving/key revocation processes do not substantially degrade the group performance as the number and speed of joining/leaving nodes increase.

Cons: The scheme did not discuss how the group manager is selected as well as the transitions of group information between the new and old group managers. The simulation tried to illustrate the impact of dynamic membership with a very small number of nodes, and the results may not be valid for a large group.

Striki and Baras [34] presented a Merkle Tree-based user authentication scheme by constructing dynamic distributed central authorities (CAs) based on Merkle trees, and then equipping these CAs with two key generation protocols: 2^d- Octopus, and Tree-

based Group Diffie-Hellman (TGDH)-based 2^d- Octopus. It has been emphasized that incorporating user authentication and key distribution algorithms in a collaborative manner into SGC yields a scalable and efficient key management protocol in MANETs.

Pros: The modified Merkle tree-based scheme with Tree-based Group Diffie-Hellman (TGDH)-based 2^d- Octopus has lower communication and processing overheads than that with 2^d- Octopus and another existing protocol, one-way function tree (OFT) as the size of the group increases.

Cons: The scheme did not discuss how this integration of authentication and key distribution could better protect SGC against various threats, such as DoS and collusion attacks.

Balachandran *et al.* [35] proposed a key agreement scheme for SGC over MANETs, referred to as the Chinese Remainder Theorem and Diffie-Hellman (CRDTH) scheme, which aims to solve two problematic issues in ad hoc environments: key serialization and absence of a central authority in MANETs. The key management in this scheme is a contributory-based GKM. All members exchange their contributed key share by using the Diffie-Hellman key exchange mechanism, and then the members independently but mutually generate the group key based on the Chinese remainder theorem (CRT).

Pros: The scheme can equally distribute the computational workloads to all members. The scheme requires only one round of broadcast to rekey the group key for a leaving process and two rounds for an initial key formation process (during a group formation) and for a joining process. No *a priori* knowledge and no member serialization are required. Highly dynamic membership is supported.

Cons: The authors only suggested how the scheme would be compromised rather than validating the security of the scheme.

Lazos and Poovendran [39] presented the routing-aware key distribution (RAwKey) problem and proposed an optimal solution that minimizes energy expenditure caused by the rekeying process in an energy-constrained wireless Ad Hoc network. The key idea is to construct an energy-efficient key distribution scheme (operating at the application layer) for SGCs over ad hoc networks by gathering cross-layer information from the physical layer (i.e., the node transmission power) and the network layer (i.e., the multicast routing tree).

Pros: The performance of the optimal energy-efficient solution for rekeying does not substantially change as the group size varies, and the cross-layer algorithm can obtain a sub-optimal solution with low complexity.

Cons: The complexity of the scheme is still rather high and the efficiency for the actual deployment remains a great challenge.

3.8 SGC over Wireless Sensor Networks

This section covers SGCs which provide security protection over wireless sensor networks.

Zhu *et al.* [31] proposed a key management protocol, called a localized encryption and authentication protocol (LEAP), for large scale distributed sensor networks. The protocol is designed based on two observations: different packet types exchanged among sensor nodes require different security services, and a single key management scheme may not be suitable for various security requirements. The proposed scheme uses four types of keys for fundamental security services, e.g., authentication and access control, to

secure communications. These four types of keys include a pair-wise key used between a sensor node and the base station, a pair-wise key used between a pair of two sensor nodes, a shared cluster key used among all sensor nodes in the same cluster, and the group key used among all sensor nodes. Thereafter, the scheme provides security services that can mitigate several attacks. For examples, authentication of one-hop broadcast communications among nodes with one-way key chains can mitigate the impersonate attack, and a timestamp is used to expire the keys to prevent the node capture and sybil attacks.

Pros: Low communication overheads; the scheme is energy efficient.

Con: The scheme did not discuss the power consumption of sensor nodes in deploying some of the proposed security mechanisms.

Yu and Guan [32] proposed a group-based key pre-distribution scheme by partitioning the network into hexagonal grids with a specified size. Nodes are then divided into groups, and each group is placed into a grid in such a way that the number of neighbors of a node is minimized, thereby reducing the power consumption. The scheme classifies communications of sensor nodes into two types: inter-group and in-group. The secret matrix G is shared by all groups, and each group is distinctively assigned a secret matrix A. Each node selects correspondingly a column from the matrix G and a row from the matrix A. Thus, two nodes in the same group can compute the pair-wise key used to secure in-group communications. Furthermore, a number of secret matrices B's are selectively assigned to groups in such a way that any two neighboring groups share a portion of the secret matrices. Then, the two neighboring groups determine which of the shared secret matrices B's they share in order to generate the shared keys. Upon the key

generation, two nodes in the neighboring groups mutually agree on which rows will be selected from these previously selected B matrices. Thus, they can compute the pair-wise key used to secure inter-group communications.

Pros: The scheme provisions a high degree of connectivity, which is defined as the fraction of the size of the largest connected components over the size of the entire sensor network. Furthermore, the connected components define a graph in which any two nodes can always find a route between them. The scheme incurs a low storage overhead, and offers a better safeguard against node capture attacks as compared to several existing schemes.

Cons: The optimal grid size may not be precisely determined, thus possibly resulting in two incidents: the inter-group keys may not be generated if the grid size is too small, and the power consumption is relatively high if the grid size is too large. The computational and time complexities might be substantial.

Zhang and Cao [33] proposed a set of pre-distributed and local collaboration-based group rekeying (PCGR) schemes to mitigate the node capture attack and the DoSS attack. The basic-PCGR (B-PCGR) scheme was first proposed with the assumption that the future group keys can be preloaded into the sensor nodes before deployment. Thus, the future keys must be protected by encryption with some polynomials, which are kept by some 1-hop away neighboring nodes. Thus, the B-PCGR scheme requires each sensor node to collaborate with each other to retrieve their encrypted future keys, and to detect and protect themselves against any attempt to compromise nodes. However, the B-PCGR scheme has two limitations: an attacker can retrieve the polynomials by searching only 1-hop away neighboring nodes of the victim, and to successfully stage an attack requires

compromising only a small number of 1-hop neighboring nodes. To overcome the first limitation, the Cascading-PCGR (C-PCGR) scheme is proposed to distribute the polynomial shares to 2- or 3-hop away neighboring nodes. To overcome the second limitation, the random variance-based PCGR (RV-PCGR) scheme is proposed to add "random variance" numbers to the polynomials to strengthen the polynomials in order to make them more difficult for the attacker to retrieve the encrypted future keys.

Pros: The schemes can effectively protect SGC against the node capture and DoSS attacks.

Cons: The rekeying is very limited owing to a limited number of reusable future keys. *A priori* knowledge of group operations (e.g., a set of future keys, key-generating function, and group joining/leaving processes) is required, and thus any real-time adaptation of these group operations cannot be performed online. The collaboration of nodes is required, but, in many sensor networks, such collaboration may not be possible at all as well as may consume unnecessary power consumption.

Huang *et al.* [40] proposed a secure level key infrastructure for multicast (SLIMCAST) to protect data confidentiality via hop-by-hop re-encryption and to mitigate the DoS-based flooding attack through an intrusion detection and deletion mechanism. The SLIMCAST protocol divides a group routing tree into levels and branches in a cluster manner. Communications among nodes in each level in each branch of the group tree is protected by a level key such that only the local level key is rekeyed during joining and leaving processes. SLIMCAST enables an upstream node to aggregate data packets from its downstream children nodes, and then re-encrypts the aggregated packet with the level keys that this node shares with its parent nodes. The re-encrypted packet is then sent

upstream towards a base station. Furthermore, the duplicate packets from the sibling nodes will be discarded to reduce redundant bandwidth and power consumption.

Pros: Low communication overheads and power consumption. The performance does not substantially degrade as the group size increases.

Cons: The performance is degraded (i.e., high power consumption) when membership changes are massive.

Wadaa *et al.* [27] proposed an energy-efficient protocol to provision anonymity in WSNs. The protocol divides the network into clusters. Two activities are defined in each cluster: intra-cluster activity (i.e., data generation) and inter-cluster activity (i.e., data transmission). For intra-cluster activities inside a cluster, a node periodically collects and formulates the sensor readings, and then reports to the designated entity, called transaction instance manager. That manager collects all node reports and formulates the transaction instance report (TIR). For inter-cluster activities, all managers send the TIR to the sink node (i.e., a base station) through neighboring managers in a hop-by-hop manner. Then, the protocol formulates the anonymity problem, and identifies and eliminates the minimum number of nodes that cause the maximum loss of sensor readings.

Pros: Energy-efficient. The performance does not substantially degrade as the group size increases.

Cons: The scheme did not analyze or prove substantially the anonymity level per transmission.

Karlof and Wagner [28] discussed attacks that can disrupt group routing in WSNs. Reference 28 illustrates how each attack is executed, and describes the existing mechanisms in mitigating the attacks.

Note that SGCs proposed for the above three types of networks may be adopted into other types of wireless networks, but consideration of such an adaptation is beyond the scope of this dissertation.

Table 3.3 Characteristics of Security Services in SGC over Wireless Networks [2]

Services	Characteristics			Details
Group key management	Type of key management		Centralized	• A dedicated entity (e.g. a key manager) generates both long-term and short-term keys, distributes them to associated members, and maintains the key material and lists. • The security of key selection and generation is high, but the key manager carries most of workloads and becomes a point of target. • In wireless infrastructure and sensor networks, this central entity can be an access point, a base station, or a dedicated key server. • However, this centralized-based GKM scheme is not applicable to ad hoc networks.
			Partially distributed	• A group is divided into smaller sub-groups. Some members in each subgroup may temporarily function as a key-generating server. • A key manager has a reduced workload. Still, it is a point of target and the security of key generation is compromised. • The distributed-based GKM scheme is applicable to ad hoc and sensor networks, but not to wireless infrastructure networks.
			Contributory	• Without a central key server/manager, each member randomly and independently selects its contribution based on some key-generating algorithms that are agreed upon by all members during the joining process. Then, the contributions are exchanged within a group, and a shared group key is identically generated. • The security of key selection and generation is low, but there is no need for a key manager. All members equally share the workloads. • This GKM type is applicable to ad hoc networks, but not to wireless infrastructure (no center) and sensor networks (the complexity of key generation and overheads are too high).
	Key secrecies	Forward secrecy		• The forward secrecy ensures that a new joining member cannot use the new key to decrypt all messages which have been encrypted with the previous key(s).
		Backward secrecy		• The backward secrecy ensures that a leaving member cannot use the previous key(s) to decrypt all messages encrypted with the new key.
		Perfect forward secrecy		• The perfect forward secrecy ensures that a compromise of a long-term key seed which was used to generate the present short-term key(s) cannot deprive the secrecy of other previous short-term keys which have been generated by the compromised long-term key seed.
	Key independence			• A disclosure of a subset of session keys cannot deprive the secrecy of other subsets of session keys which have been generated by the same long-term key seed.
	Key serialization			• In some distributed and contributory-based GKM schemes, the key materials are selected and the group key is generated by members in an ordered sequence, as discussed in [8]. An attack on any participating member disrupts the whole process. Note that the key materials are actually key seeds used to generate keys (e.g., primes in RSA or DH) • Instead, some schemes may construct the key by other ways, i.e., broadcasting the key materials to all members or establishing a key tree, at the expense of communication and storage overheads. In addition, with insecure wireless channel, these ways may lead to service unavailability especially in ad hoc networks where membership is highly dynamic.
	Rekeying	# of rekey messages		• An average number of distributed and received messages per member (or per key manager) should be minimized.
		Length of rekey messages		• Some protocols reduce the number of rekey messages by aggregating multiple messages into a single message, which in return increases the consumed bandwidth for one transmission. Thus, the performance metric should also determine the bandwidth consumption per message in addition to the number of transmitted messages.
		Rekeying Process		• The rekeying operation should reduce or optimize the computation and time complexity of rekeying operation with respect to a group size.
		Triggering conditions	Membership	• Keys associated with the membership changes must be rekeyed to ensure the key secrecy for the remaining members.
			Periodic	• To provide a better key secrecy, the rekeying operation is invoked periodically to prevent keys from being compromised over time.
			Specified	• A system enables the rekeying operation for specified incidents, such as upon a detection of attacks or violations.
Authentication	Message authentication			• A system should require a sender to sign a message and a receiver to verify such a message signature on its authenticity as well as integrity.
	User authentication			• Users should be authenticated upon joining the group, signing the messages, or accessing group materials.
		Sender viewpoint		• A designated sender may want to multicast a message on behalf of the group to all designated receivers without revealing its real identity.
		Receiver viewpoint		• A receiver verifies whether the message has been originated by an unspecified group member (not an outsider). • OR a receiver verifies whether the message has been originated by an unspecified but designated sender (not an outsider and not designated receivers). • OR, in a possibly disputable case, a receiver verifies whether the message has been originated by the specified and designated sender.
Access control	Authorization/access control techniques			• The group resources (e.g., key seeds, keys, a member list, multicast routing tables, etc.) and group messages should be accessible only to authorized members. • The access control list can be used to determine if the member has permission to access resources and, more specifically, to which resource is accessible.
	Dynamic access control			• A system enables the member to dynamically change its request to access resources. Consequently, the system must be able to update access permissions and restrictions with additional mechanisms when the member's access privilege changes. Other systems may simply give all members a fixed access control privilege per session.
Non-repudiation	Message signature			• A system requires messages to be signed or equipped with a membership certificate so that an originator (signer) of the message can always be identified.
	Timestamp and/or sequence number			• Timestamps and/or sequence numbers can be used to limit a validation of certificate or message signature, and to prevent replay attacks.
	Revocation of certificates			• The expired certificate or misuse of certificate is revoked by the issuer and may be publicly announced in the revocation list. • For higher non-repudiation, some systems may keep the expired certificates for future verification at the expense of storage overhead.
	Characteristics of signature		Unforgeable	• A group of colluded attackers cannot generate a group signature identical to that generated by a legitimate member.
			Non-allegeable	• A group of colluded attackers cannot generate a group signature by which a group controller falsely identifies a legitimate member as an attacker.
			Linkable	• A group of colluded attackers cannot generate a valid group signature by which a group controller cannot identify the identity of any of these attackers.
			Secretive	• A member's secret elements can neither be retrieved from a group signature nor from any part of it.
Privacy and Anonymity	Unlinkability of anonymous communications		Sender	• A sender shall not be linked to its sent message to prevent attackers from learning of the message's origin.
			Receiver	• A receiver shall not be linked to the received message to prevent attackers from learning of the message's destination.
			Sender-Receiver	• The sender and receiver shall not be linked together to prevent attackers from learning of the sending and receiving ends. They are also relatively anonymous to each other.
	Type of management		Centralized	• A system relays messages through a trusted anonymous entity to hide identities of the sender and receiver.
			Distributed	• A system relays messages through a group of anonymous entities to hide the identities by various means such as encapsulating messages.
Secure routing	Management			• A system can establish and maintain routing-related information in a centralized or distributed manner.
	Prevention			• Updating routing information (e.g., a membership status and routing paths) must be restricted to authorized members. • A new routing path should be tested such that any routing black hole and loop are entirely eliminated.

3.9 Open Challenges

Here, this work outlines some challenges that should be tackled, and define future research directions on SGC over wireless networks.

1. *Integration of security services.* As illustrated in Table 3.2, most attacks can be greatly mitigated by the fundamental security services. Thus, it is still the greatest challenge to design an energy-efficient scheme that integrates more security services to satisfy various security requirements, particularly, authentication, access control, and non-repudiation (via group signatures), without incurring significant overheads.
2. *Deployment of SGC in heterogeneous wireless networks.* With a higher demand for more functionalities in wireless devices, a SGC scheme should be able to be deployed in heterogeneous wireless networks, including cellular networks, Wireless LANs, wireless ad hoc networks, and wireless sensor networks, and to support communications over a hybrid of wireless and wired networks.
3. *Optimization on group performance with respect to overheads and limited resources.* To be efficient, a scheme should optimize group performance subject to overheads (communication, processing, and storage) and limited resources (i.e., memory, bandwidth, and power supply).
4. *Extension to IPv6 wireless networks.* An IPv6 wireless network seems to be a promising next generation network, and some works are addressing the end-to-end built-in security. Future research on SGC is engineered to support IPv6 wireless networks.

3.10 Conclusion

The number of applications of group communications over wireless networks has been steadily increasing such as group-oriented military systems (on-the-field commander conference over wireless devices) and education systems (teacher lecturing in a distant learning classroom). However, communications over wireless channels is, by nature, insecure and easily susceptible to various kinds of attacks. Many existing works have attempted to incorporate security into such communications.

To better understand SGC over wireless networks, this work presents known attacks that can severely disrupt or even shut down group communications in wireless networks. Then, necessary security requirements are discussed and fundamental security services are provided to meet these requirements and safeguard the communications against these attacks. This work has demonstrated that several attacks can be prevented and mitigated by proposed security services. This work has also reported several existing works on SGC over three types of wireless networks: wireless infrastructure networks, mobile ad hoc networks, and wireless sensor networks, as summarized in Table 3.4. With respect to limited computation capability and scarce wireless channels, these works basically attempt to reduce communication and processing overheads, and to fend off some particular attacks. To complete the survey on SGC over wireless networks, this work has presented some open challenges that still need to be overcome.

Table 3.4 Comparison of SGC over Wireless Networks [2]

Schemes Characteristics	Wireless Infrastructure Networks				Mobile Ad Hoc Networks				Wireless Sensor Networks				
	[5]	[12]	[1]	[13]	[11]	[9]	[10]	[14]	[6]	[7]	[8]	[15]	[2]
Key Management	Hierarchical-based	Topology-matching tree	Area-based and Batch rekeying	N/A[1]	Ad Hoc Group Key (AGK)	Tree-based group Diffie-Hellman (TGDH)	Chinese Remainder and Diffie Hellman	Routing-aware key distribution	Cluster-based keys	Group-based key	Locally group-based key and Key pre-distribution	Cluster-based Tree (Level and branch)	N/A
Authentication	N/A	N/A	User auth.	Key and Message auth.	Certificate-based PK[2] auth.	ID-based User auth. and Merkle tree-based data auth.	N/A	N/A	Source and Message one-way key chain-based and Challenge-response auth.	N/A	N/A	MAC signature and One-way sequence number	N/A
Authorization/ Access Control	N/A	N/A	Location-based	Packet filtering	certificate	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Accounting/ Non-repudiation	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Yes	N/A
Message Integrity and confidentiality	Yes	N/A	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	Yes	N/A
Privacy/ Anonymity	N/A	N/A	Privacy of user's location	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Yes (virtual infrastructure)
Survivability/ Availability	N/A	N/A	N/A	N/A	Yes (quick recovery)	Yes	N/A	N/A	Yes	Yes	Yes	N/A	Yes
Attack prevention[3]	N/A	N/A	Eavesdropping	DoS, Identity, Replay, Source address spoofing	Message modification, Replay	Impersonate, Collusion	N/A	N/A	Wormhole, Sinkhole, Sybil, DoS, Replay, Insider	Node capture	Node capture, Eavesdropping, DoSS	Node capture, Sybil	DoS, Traffic analysis
Reducing communication overheads	Yes	Yes (using locality to reduce comm. complexity)	Yes (with HYBRID scheme)	N/A	Yes (using locality to reduce comm. complexity)	Yes	N/A	Yes	Yes (using locality to reduce comm. complexity)	N/A	Yes	Yes	Yes
Reducing processing overheads	Yes	Yes	Yes (with HYBRID scheme)	N/A	N/A	Yes	Yes	N/A	Yes	N/A	Yes	Yes	N/A
Handling high mobility	Yes	Yes	Yes	Yes	Yes	N/A	Yes	Yes	N/A	N/A	N/A	Yes	N/A
Handling handoffs	Yes (but needs back channels)	Yes	N/A	Yes	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Steady performance vs. group size	N/A	Yes	N/A	N/A	Yes	Yes	N/A	Yes	N/A	N/A	N/A	Yes	N/A
Scalable	Yes	Yes	N/A	N/A	Yes	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes
Energy-efficient	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Yes	N/A	N/A	N/A	Yes	Yes

Note 1: N/A = Information not available about the characteristic OR the characteristic is not likely possible or not applicable.
2: PK = Public Key.
3: Only specified attacks discussed in the respective references are listed here even though each of these schemes may mitigate other attacks as well.

CHAPTER 4

ADAPTIVE TRUST AND REPUTATION SYSTEM

4.1 Objective

This section introduces the trust and reputation system with two objectives: to propose the resilient trust and reputation system that can be implemented in different network environments; to analyze security concerns in many aspects including an attack classification, an illustration of attack scenarios, and an introduction of defense mechanisms responding dependently on every attack scenarios.

4.2 Introduction of Trust and Reputation System

A trust and reputation system becomes an increasingly important research area and is implemented in many commercial websites, in particular auction websites, which allow a customer to rate other customers' auction behaviors. In general, all parties are evaluated for different reasons: a seller lies about the products he is selling or dishonestly blames the buyer for some bad products; a buyer lies about the services or products he unsatisfactorily received and he exaggerates the bad reviews about the seller; a group of buyers colludes to give bad reviews to a specific seller such that the seller becomes very untrustworthy; a buyer and a seller collude to exaggerate very good reviews to each other such that they become very trustworthy without actually performing proper actions.

4.3 The Adaptive Trust and Reputation System

This section discusses the processes of the trust and reputation system.

4.3.1 Definitions of Trustworthiness, Trust, and Reputation

There are many definitions regarding to the taxonomy of existing trust and reputation systems, but in general, trust relationship among network entities can be established based on two types of experiences: direct experience and indirect experience. In most trust and reputation systems, the direct experience can be obtained from self observations and the indirect experience can be obtained from other network entities' direct experience toward the entity of interest. However, in this dissertation, direct experience is obtained from self observations and other network entities' observation on transactions that the node, referred to as the originator node, generates or establishes. The indirect experience is simply a report of the trustworthiness value of the entity of interest that is known to the reporting node from previous transactions. The transaction in this dissertation includes all network activities depending on applications into which the trust and reputation system has been applied or integrated. This dissertation assumes that any transaction is done in a certain period of time, referred to as a session. If the trust and reputation system is applied into communication networks, the transaction might be the transmission of control and data packets generated by a specified application or protocol. All packets are transmitted within a specified time period per connection. If the connection is continuous and lasts for a long period of time, the session can be taken by a cycle ranged in a certain period of time.

Direct experiences are observations done by the originator itself or observations done by other network entities who report the findings to the originator. To perform observations, this dissertation does not specify how the observation can be done. The dissertation, however, suggests that the observation may be done dependent on the network environment into which the trust and reputation system is being integrated. The centralized networks may create a monitoring entity designed to make observations on other network entities within a close neighboring area. The monitoring entity becomes a central authority to collect, manage, and interact all observation findings. The distributed networks may rely on all network entities to make observations among themselves. The problem arises in the distributed networks in the case that the observations may be limited owing to a geographical matter and there is no coordinating entity to collect, store, or interact all observation findings. The challenge to deal with observation mechanisms and associated information is still a daunting task in the distributed networks, especially in wireless ad hoc networks or vehicular ad hoc networks.

In this dissertation, direct experiences are the observation findings from the current or most recent session. The direct experiences can be referred to as *Trust*. Meanwhile, indirect experiences are the results taken into calculation with observation findings from the previous sessions. The indirect experiences can be referred to as *Reputation* in this dissertation.

Trustworthiness is a combination of trust and reputation in a proportional manner. The significance of trust weighs more than that of reputation. The trustworthiness of a network entity is defined as:

$$Trustworthiness = (\alpha \text{ x } Trust) + (\beta \text{ x } Reputation),$$

where α and β are weight factors determining the significance of trust and reputation in the calculation of trustworthiness, respectively. In addition, the weight factors obey the following rule:

$$\alpha + \beta = 1$$

When the new trustworthiness is being re-evaluated, the old trustworthiness is used as a reference for the update as follows:

$$New\ Trustworthiness = \rho_1\ [(\alpha \text{ x } Trust) + (\beta \text{ x } Reputation)] + (\rho 2 \text{ x } Old\ Trustworthiness)$$

This dissertation use ρ_1 and ρ_2 as factors to determine the timing of maintaining the trustworthiness. The idea is that the new trustworthiness should receive a greater attention than the old value because it freshly reflects the most recent experience.

Let $T_{X,Y}^{Si}$ be the trust value of node X known to node Y. Node Y has observed on the specific packets transmissions T_i (during the i^{th} session) that node X has performed forwarding packets,

$R_{X,Y}^{Si}$ be the reputation value of nodes X known to node Y. Node Y has observed on its own specific packets transmissions T_i (during the i^{th} session) that node X has performed forwarding packets,

$TW_{X,Y}^{Si}$ be the trustworthiness value of node X known to node Y that is evaluated after the i^{th} session, and can be calculated as follows:

$$TW_{X,Y}^{Si} = \alpha \times T_{X,Y}^{Si} + \beta \times R_{X,Y}^{Si}$$

$$R_{X,Y}^{Si} = \sum_{Z \in M} \left(TW_{X,Z}^{Si} \times TW_{Z,Y}^{Si-1} \right),$$

where $TW_{X,Y}^{Si}, T_{X,Y}^{Si}, R_{X,Y}^{Si}, TW_{X,Z}^{Si} \in [0,1]$ and M is the set of all observing nodes.

The trustworthiness is uni-directional; the trustworthiness of node X known to node Y may be different from the trustworthiness of node Y known to node X. This can be depicted as follows:

$$TW_{X,Y}^{Si} \neq TW_{Y,X}^{Si}.$$

When the trustworthiness of a node is evaluated after the i^{th} session, the trustworthiness evaluated after the i^{th} session is calculated as follows:

$$TW_{X,Y}^{Si} = \rho_1 \left(\alpha \times T_{X,Y}^{Si} + \beta \times R_{X,Y}^{Si} \right) + \rho_2 \left(TW_{X,Y}^{Si-1} \right).$$

Thereafter, the system performs the update process to ensure that any possible error or malicious update on the trustworthiness value is prevented or mitigated.

4.3.2 Trustworthiness Initialization Process

Depending on network environments that the trust and reputation system is integrated into, various admission or initialization mechanisms can be deployed to handle the joining request from a new node. If the network is centralized, there is usually a central authority to check the new node's identity and provide the new node the joining permission. Based on information given by the new node, a digital certificate may be calculated and issued to enhance the authentication service. Without a central certificate

authority, a distributed digital certification protocol can be adopted to calculate and issue the certificate. If the network is distributed where there is no central authority, the new node may contact a node who is intermittently assigned to admit new nodes or may simply broadcast all nodes in the network to get instructions. Although the certification is assumed to be securely available, the processes of generating, distributing, validating and revoking are beyond the scope of this dissertation. Furthermore, this chapter does not intend to suggest how the new node is admitted, but instead assumes that such admission mechanisms are already in place. The dissertation simply lays out how the trustworthiness value is initialized and constantly re-evaluated.

A node creates an entry in the node's trustworthiness-based table or cache, and assigns the initial trustworthiness value to the new node. The initial value should be the same to all new nodes and has a minimum value to mitigate a "Re-enter" attack. The detail of the re-enter attack is given in the following section. Furthermore, the initial value should be set differently according to the implementation. Some networks may require the new nodes to take a slow pace to update the trustworthiness value, so they may set the initial value very low, perhaps nearly close to the blacklist threshold.

In addition to a newly joined node, an initial trustworthiness value (*INIT_TW*) is assigned to an unknown node which has never been in the trustworthiness-based table. That means, an unknown node is treated as a node newly joining the network.

The node may send out the trustworthiness request message regarding the new node. Any node which receives this request message and has knowledge about this new node replies the request message.

$$TW^{Si}_{New\_n,Y} = \rho_1\left(\alpha \times T^{Si}_{New\_n,Y} + \beta \times R^{Si}_{n,Y}\right)$$

$$T^{Si}_{New\_n,Y} = INIT\_TW$$

$$R^{Si}_{n,Y} = \frac{\sum_{n \in N}\left(TW^{Si}_{X,n} \times TW^{Si-1}_{n,Y}\right)}{\sum_{n \in N} TW^{Si-1}_{n,Y}},$$

where n represents neighbor nodes that the trustworthiness request messages inquired about the new node, *New_n*, sent by the node Y, and N is the set of nodes replying the request message.

The expiry timer in the trustworthiness-based table is set to the same period of time for each entry.

4.3.3 Trustworthiness Monitoring Process

This work does not specify the monitoring process as who monitors and how to monitor, as how to detect and identify an error, as whether the error is a genuine performance problem or a malicious security problem, and as how frequent the monitor is taken place. However, the monitoring mechanism should be dependent on network environments. There are simple suggestions for the monitoring mechanism that, during the transmission session, there can be four parties monitoring the current transmission session as follows.

1. The source node makes a self-observation on close-distance neighbors' behaviors on the current transmission session.
2. The destination node monitors behaviors of all intermediate nodes, as well as the source node, throughout the current transmission session.

3. An intermediate node monitors behaviors of some other intermediate nodes, whose distances are within a close distance, throughout the current transmission session.
4. An observing node whose location is within a close distance makes an observation on all intermediate nodes' behaviors on the current transmission session.

According to certain criteria, different negative behaviors are rated with different scores. With a promiscuous mode, a node can detect behaviors of its close-distance neighbors. All nodes can be publicly given certain criteria for assessing the behaviors.

Criteria	Weight of Criteria	Classification	Error Notification
Criteria 1	Wc_1	Critical	Immediate
Criteria 2	Wc_2	Critical	Immediate
Criteria …	…	…	…
Criteria n	Wc_n	Non-critical	Delayed

The weight of each criterion is lower than one for any non-critical error or equal to one for any critical error; $W_{C_j}\begin{cases}<1 & non-critical\\ =1 & critical\end{cases}, j \in J$, where J is the set of all criteria. To reduce the number of error notification messages, a node may not need to send error messages every time an error is detected. The proposed system enables two responding measures: immediate and delayed notification, depending on the severity of the error.

- If the error matches the criterion regarded as critical, i.e., threatening the disruption of the system or violating the integrity of data contents, the monitoring

node must generate an error notification message and immediately send it out individually to the source and destination nodes.

- If the error matches the criterion regarded as non-critical, i.e., having a drop rate of 5%, the monitoring node can delay the notification by aggregating several more evidences within a certain time period and reports the aggregated message via piggybacking with other packets, such as ACK packets, if such packets are available.

4.3.4 Error Detection Process

The error detection process starts with the performance analysis that may be interacted with human network operators to adaptively determine which criteria demonstrate the error event both in the performance and security issue. Once the criteria are identified, the weigh is proportionally associated with each criterion to determine how serious that criterion should be taken into calculation. This dissertation does not offer the details of criteria because they are dependent of the network into which the trust and reputation system is integrated. There are many research works that offer many performance analyses for each network, and some or all of criteria that are used in those works can be adopted in the trust and reputation system.

If the criteria are not met, it is reasonable to presume that no error has occurred. The weight and classification of criteria can be adjusted, given the feedback from the performance analysis. However, if new criteria emerge, the trust and reputation system can add the criteria to reflect the dynamic need of the network. The criteria are comprised as the behavior policy. As depicted in Figure 4.1, the observing and detection mechanisms are not considered in this work, but they should vary depending on the

network environment into which the trust and reputation system is integrated. With criteria and their associated weighs, if the behavior or error is determined as critical (i.e., some security violation issues), the error notification is sent out immediately to the source and destination. The source may apply some attack prevention mechanisms to deal with the problem. If the behavior is deemed as non-critical (i.e., performance issues), the observation results are aggregated for the whole session, and the observation report is sent out to the source node after the session ends. All bad behaviors are recorded in some kind of an error cache.

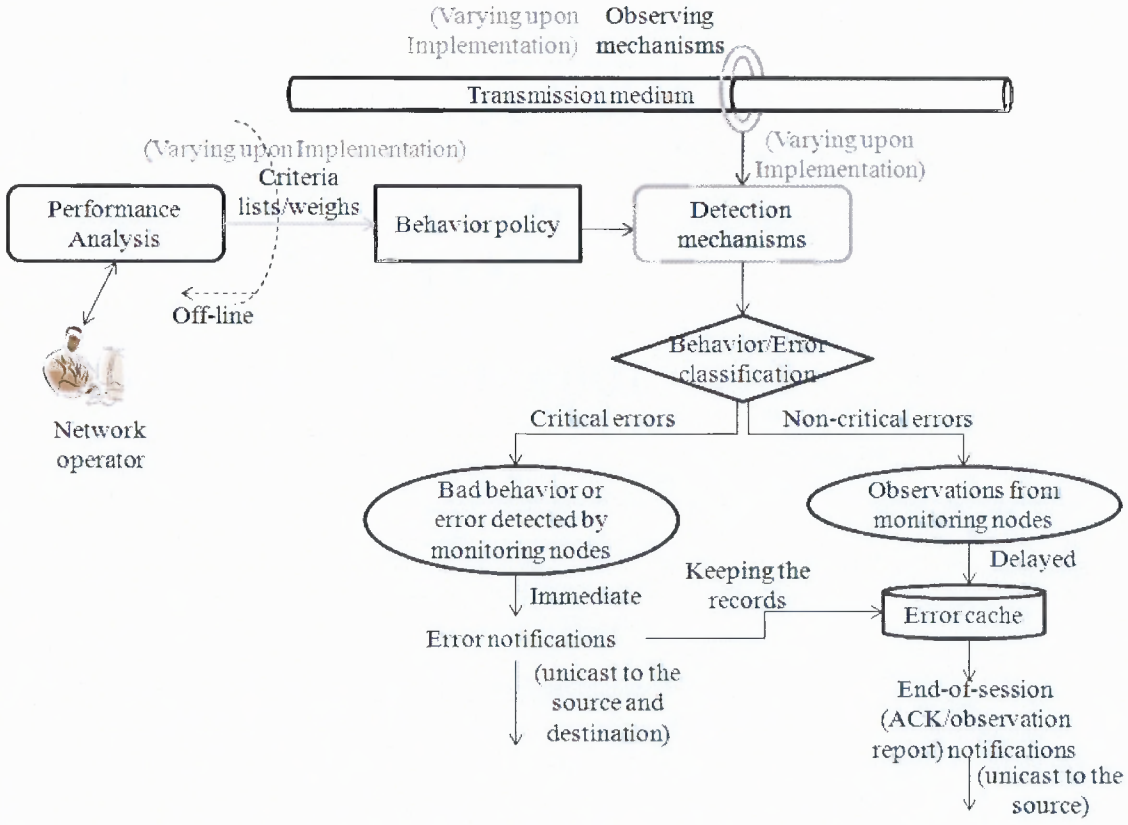


Figure 4.1 An illustration of the error detection process.

4.3.5 Trustworthiness Request Process

The trustworthiness request process can be triggered by three scenarios:

1. When the validation timer for a trustworthiness entry expires. If the timer of an entry has expired, the node can independently and individually choose
   - whether the trustworthiness request process should be executed to obtain newer trustworthiness information of that entry, or
   - whether that entry should be discarded to reserve memory space for future use.

   Each node may have different memory space resources to store a different number of entries in its trustworthiness table. If the node has less memory spaces, it can delete the entries which have expired without triggering the trustworthiness request process.
2. When the blacklist notification is received. If the node receives a blacklist notification from its neighbors, it checks whether the blacklisted node is in the trustworthiness-based routing table. If that is confirmed, the node independently and individually decides whether the trustworthiness request and re-evaluate processes should be executed to reevaluate the blacklisted node. In some networks where security is far more important than network overheads, all nodes can be mandated to initiate the trustworthiness request and re-evaluation processes upon the receipt of the notification.
3. When this is the first interaction with an unknown node. Since an unknown node is initially assigned the initial trustworthiness value, it is reasonable to expand the

knowledge regarding this new node by sending out the trustworthiness request message.

In the proposed system, there are two types of the trustworthiness request processes:

- Local trustworthiness request (LTR) process. LTR requires a node to broadcast the local trustworthiness request (LTReq) message to all of its close-distance neighbors. The range of broadcasting the message can be limited to one or a few hops away, by setting the value of Time-To-Live (TTL) in the IP header.
- Global trustworthiness request (GTR) process. GTR requires the node to multicast or broadcast the global trustworthiness request (GTReq) message. To reduce the communication overhead, i.e., the number of trustworthiness request messages, the node aggregates all nodes of interest in one message. A node should be limited to a certain number of trustworthiness request messages per a certain period of time, depending on the network implementation. To further reduce the communication overhead, a node multicast the aggregated message to a limited number of receivers, i.e., the first m entries in the trustworthiness cache (sorted by the highest trustworthiness values).

The generation of either GTReq or LTReq messages should automatically prompt the reset of the TREQ timer. The GTReq message is broadcast to all nodes in the network, thus reaching a larger pool of inquired nodes, and so the TREQ timer is set to a longer time period. Once the TREQ timer expires, all reply messages are immediately discarded.

The condition whether to trigger either LTR or GTR process depends on the following three factors:

1. The responding time. GTR sets the timer the longer time period in order to gather more reply messages, while LTR sets the timer the shorter time period. This factor may outweigh other factors in cases when the validation timer for a trustworthiness entry expires and when the proposed system is integrated to delay-sensitive networks. In these cases, LTReq messages provide information quicker than GTReq.
2. The numbers of LTReq messages and GTReq messages. This factor may outweigh other factors in some cases such as when the performance (in terms of overheads) is more important than security (in terms of attack mitigation) and when the blacklist notification is received. In the former case, a node is required to first inquire close neighbors rather than nodes located multiple hops away. The reason is that neighbors within the close proximity may experience or observe behaviors or errors similarly especially when those are related to network performance issues, i.e., long delay, congestion, and link failures. Thus, LTR gathers information that is more related to the inquiring node. In the later case, GTReq messages are sent to mitigate the possible incident that the inquiring node is surrounded by malicious nodes, thus being manipulated through LTReq messages.
3. The trustworthiness level of the sender known to the receivers. Once the receiver receives the trustworthiness request message, it checks the trustworthiness value of the sending (inquiring) node. If the sending node has

low trustworthiness value, the receiver may discard the request. This factor may outweigh other factors in some cases such as when the new node joins in and when this is the first interaction with an unknown node. In the former case, LTReq should be sent because GTReq messages are likely to be discarded due to the fact that the new node has a low trustworthiness value. In the later case, GTR can provide a larger amount of information than LTR because it is more likely that the close neighbors within the range of LTReq messages may also never communicate with the unknown node.

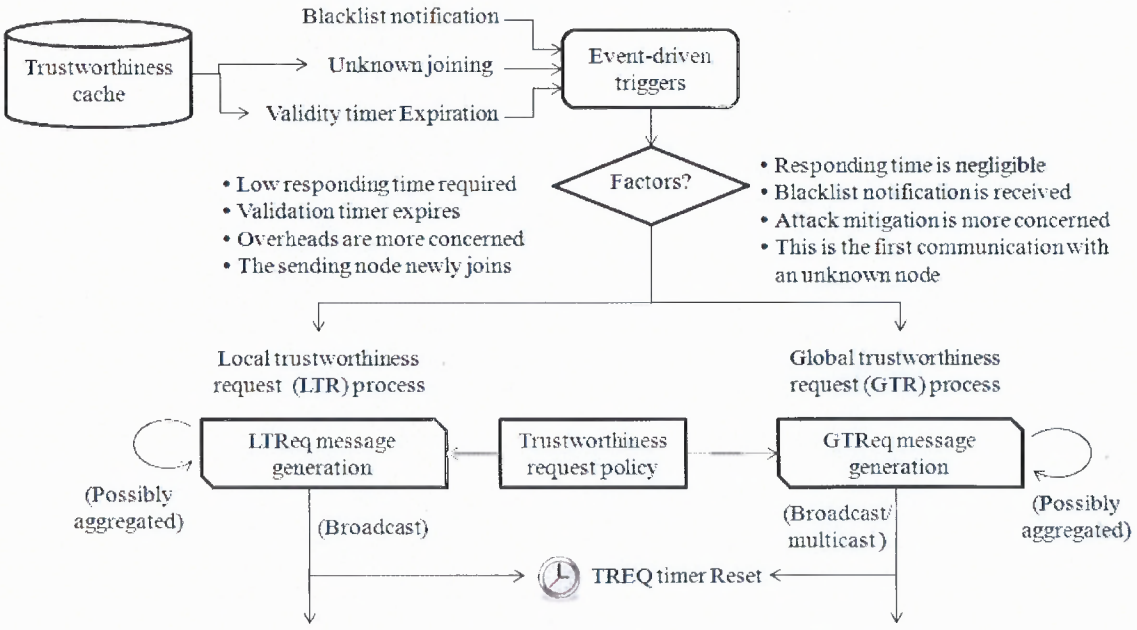


Figure 4.2 An illustration of the trustworthiness request process.

4.3.6 Trustworthiness Re-evaluation Process

The trustworthiness value of all intermediate nodes involved in network activities, i.e., data and control packets transmission and trustworthiness request, is calculated after a transmission session. Let S_i be the current transmission session and S_{i-1} be the previous transmission session. All nodes individually start re-evaluating the behaviors of other nodes involved in the activity based on given criteria and error notifications. In this chapter, the network activities are data transmission and trustworthiness request for the sake of simplicity.

1) Trustworthiness reevaluation after data transmission

Assume all intermediate nodes are in the set of M_I and any error-reporting nodes are in the set M_E, the sending and receiving nodes are represented as X and Y, respectively. Any node keeps track of how many times the error notification messages sent by the same observing node are proven to be faulty. The observing node may be regarded as malicious if the ratio of the number of faulty error notification messages to the number of correct error notification messages exceeds a certain number, called the false notification threshold. Each node involved in the data transmission reevaluates each other nodes one by one as depicted in Table 4.1:

Table 4.1 Re-evaluation Scenarios

		Reevaluated node			
		Sending node	Receiving node	Intermediate (forwarding) node(s)	Observing node(s)
Reevaluating node	Sending node	Self	*T* – self observed *R* – combined error reports	*T* – self observed, if possible *R* – combined/ received error reports	*T* – no *R* – received error reports
	Receiving node	*T* – self observed *R* – combined error reports	Self	*T* – self observed, if possible *R* – combined/ received error reports	*T* – no *R* – received error reports
	Intermediate (forwarding) node(s)	*T* – self observed, if possible *R* – received error reports	*T* – self observed, if possible *R* – received error reports	*T* – self observed, if possible *R* – received error reports	*T* – no *R* – received error reports
	Observing node(s)	*T* – self observed, if possible *R* – received error reports	*T* – self observed, if possible *R* – received error reports	*T* – self observed, if possible *R* – received error reports	*T* – no *R* – combined/ received error reports

A. Case 1, a reevaluating node is a sending node. There are three possible reevaluations as follows:

- A reevaluated node is the receiving node. The trust calculation gathers information from self observations, and the reputation calculation gathers error notification messages received by any nodes (i.e., observing nodes, intermediate nodes, and, the receiving node itself—in case of maliciously denying the receipt of packets). The new trustworthiness for the receiver is calculated as follows:

$$New\_TW_{Y,X}^{Si} = \rho_1 \left\{ \alpha T_{Y,X}^{Si} + \beta R_{Y,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{Y,X}^{Si-1} \right\}, \text{ where } e \in M_E$$

- A reevaluated node is an intermediate node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from multiple observing nodes. The new trustworthiness for the intermediate node is calculated as follows:

$$New\_TW_{d,X}^{Si} = \rho_1 \left\{ \alpha T_{d,X}^{Si} + \beta R_{d,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{d,X}^{Si-1} \right\}, \text{ where } d \in M_I, e \in M_E$$

- A reevaluated node is an observing node, say node *E*. The trust calculation may be neglected since the sending node is likely not to observe the observing node at that time, but the reputation calculation gathers multiple error notification messages regarding the same error and determines whether the node *E* gives out the notification accurately. The new trustworthiness for the observing node is calculated as follows:

$$New\_TW_{E,X}^{Si} = \rho_1 \left\{ \beta R_{E,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{E,X}^{Si-1} \right\}, \text{ where } e \in M_E \cap \{E\}$$

B. Case 2, a reevaluating node is a receiving node. There are three possible reevaluations as follows:

- A reevaluated node is the sending node. The trust calculation gathers information from self observations, and the reputation calculation gathers the received error notification messages. The new trustworthiness for the sender is calculated as follow:

$$New\_TW_{X,Y}^{Si} = \rho_1 \left\{ \alpha T_{X,Y}^{Si} + \beta R_{X,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{X,Y}^{Si-1} \right\}, \text{ where } e \in M_E .$$

- A reevaluated node is an intermediate node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from multiple observing nodes. The new trustworthiness for the intermediate node is calculated as follows:

$$New\_TW_{d,Y}^{Si} = \rho_1\left\{\alpha T_{d,Y}^{Si} + \beta R_{d,e}^{Si}\right\} + \rho_2\left\{Old\_TW_{d,Y}^{Si-1}\right\}, \text{ where } d \in M_I, e \in M_E.$$

- A reevaluated node is an observing node, say node E. The trust calculation may be neglected since the receiver is likely not to observe the observing node at that time, but the reputation calculation gathers multiple error notification messages regarding the same error and determines whether the node E gives out the notification accurately. The new trustworthiness for the observing node is calculated as follows:

$$New\_TW_{E,Y}^{Si} = \rho_1\left\{\beta R_{E,e}^{Si}\right\} + \rho_2\left\{Old\_TW_{E,Y}^{Si-1}\right\}, \text{ where } e \in M_E \cap \{E\}.$$

C. Case 3, a reevaluating node is an intermediate (forwarding) node, say node D. There are four possible reevaluations as follows:

- A reevaluated node is the sending node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from multiple observing nodes. The new trustworthiness for the sender is calculated as follows:

$$New\_TW_{X,D}^{Si} = \rho_1\left\{\alpha T_{X,D}^{Si} + \beta R_{X,e}^{Si}\right\} + \rho_2\left\{Old\_TW_{X,D}^{Si-1}\right\}, \text{ where } e \in M_E.$$

- A reevaluated node is the receiving node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from multiple observing nodes. The new trustworthiness for the receiver is calculated as follows:

$$New\_TW_{X,D}^{Si} = \rho_1\left\{\alpha T_{X,D}^{Si} + \beta R_{X,e}^{Si}\right\} + \rho_2\left\{Old\_TW_{X,D}^{Si-1}\right\}, \text{ where } e \in M_E.$$

- A reevaluated node is another intermediate node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from multiple observing nodes. The new trustworthiness for the intermediate node is calculated as follows:

$$New\_TW_{d,D}^{Si} = \rho_1\left\{\alpha T_{d,D}^{Si} + \beta R_{d,e}^{Si}\right\} + \rho_2\left\{Old\_TW_{d,D}^{Si-1}\right\}, \text{ where } d \in M_I, e \in M_E.$$

- A reevaluated node is an observing node, say node E. The trust calculation may be neglected since the receiver is likely not to observe the observing node at that time, but the reputation calculation gathers multiple error notification messages regarding the same error and determines whether the node E gives out the notification accurately. The new trustworthiness for the observing node is calculated as follow:

$$New\_TW_{E,D}^{Si} = \rho_1\left\{\beta R_{E,e}^{Si}\right\} + \rho_2\left\{Old\_TW_{E,D}^{Si-1}\right\}, \text{ where } e \in M_E \cap \{E\}.$$

D. Case 4, a reevaluating node is an observing node, say node *E*. There are four possible reevaluations as follows:

- A reevaluated node is the sending node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from other observing nodes. The new trustworthiness for the sender is calculated as follows:

$$New\_TW_{X,E}^{Si} = \rho_1 \left\{ \alpha T_{X,E}^{Si} + \beta R_{X,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{X,E}^{Si-1} \right\}, \text{ where } e \in M_E .$$

- A reevaluated node is the receiving node. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from other observing nodes. The new trustworthiness for the receiver is calculated as follows:

$$New\_TW_{Y,E}^{Si} = \rho_1 \left\{ \alpha T_{Y,E}^{Si} + \beta R_{Y,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{Y,E}^{Si-1} \right\}, \text{ where } e \in M_E .$$

- A reevaluated node is an intermediate node, say node *D*. The trust calculation gathers information from self observations if that is possible, and the reputation calculation combines one or more error notification messages from other observing nodes. The new trustworthiness for the sender is calculated as follows:

$$New\_TW_{X,E}^{Si} = \rho_1 \left\{ \alpha T_{X,E}^{Si} + \beta R_{X,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{X,E}^{Si-1} \right\}, \text{ where } e \in M_E .$$

- A reevaluated node is another observing node, say node *E*. The trust calculation may be neglected since the receiver is likely not to observe the observing node at that time, and the reputation calculation combines one or more error notification

messages from other observing nodes. The new trustworthiness for the observing node is calculated as follows:

$$New\_TW_{X,E}^{Si} = \rho_1 \left\{ \alpha T_{X,E}^{Si} + \beta R_{X,e}^{Si} \right\} + \rho_2 \left\{ Old\_TW_{X,E}^{Si-1} \right\}, \text{ where } e \in M_E .$$

The key idea is that the new possible trustworthiness value is an old trustworthiness value plus a change in the trustworthiness value based on behaviors that are recently observed.

Trust and reputation values are calculated from the criteria as follows:

$$T_{n,reevaluating\_node}^{Si} = 1 - \frac{\sum_{j \in J'} W_{C_j}}{\sum_{j \in J'} C_j} \text{ and } R_{n,reevaluated\_node}^{Si} = 1 - \frac{\sum_{j \in J'} W_{C_j}}{\sum_{j \in J'} C_j},$$

where $W_{Ci} \in [0,1]$, $e \in M_E$, and J' is the set of reported errors.

- If the reevaluating node observes no error, the trust value of the reevaluated node, in this case *n*, becomes one. Similarly, if one critical error is observed, the trust value for that node becomes zero. Non-critical errors yield lower trust values.
- If the reevaluating node receives no error notification message, the reputation value of the reevaluated node, in this case *n*, becomes one. Similarly, if one critical error is observed, the reputation value for that node becomes zero. Non-critical errors also yield lower reputation values.
- Summarily, with no error observed or reported, it is calculated as

$$New\_TW_{reevaluated\_node,reevaluating\_node}^{Si} = \rho_1 \{1\} + \rho_2 \left\{ Old\_TW_{reevaluated\_node,reevaluating\_node}^{Si-1} \right\}.$$

-

- Otherwise, the new trustworthiness value is calculated as

$$New\_TW^{Si}_{reevaluated\_node,reevaluating\_node} = \rho_1 \left\{ \begin{matrix} \alpha T^{Si}_{reevaluated\_node,reevaluating\_node} \\ +\beta R^{Si}_{reevaluated\_node,reevaluating\_node} \end{matrix} \right\} + \rho_2 \left\{ Old\_TW^{Si-1}_{reevaluated\_node,reevaluating\_node} \right\}$$

2) Trustworthiness reevaluation after the trustworthiness request

After the node sent and received the LTREQ and GTREQ messages, it reevaluates the trustworthiness values of the nodes of interest as follows:

- The reputation calculation combines all trustworthiness replies regarding the node of interest, in this case, *n*.

$$New\_TW^{Si}_{n,X} = \rho_1 \left\{ \beta R^{Si}_{n,e} \right\} + \rho_2 \left\{ Old\_TW^{Si-1}_{n,X} \right\},$$

where $e \in M_E$ (set of replying nodes) and $R^{Si}_{n,e} = \left[\sum_{e \in M_E} (TW^{Si}_{n,e} \times TW^{Si-1}_{X,e}) \right]$.

The following Figure 4.3 illustrates the trustworthiness reevaluation process as previously described.

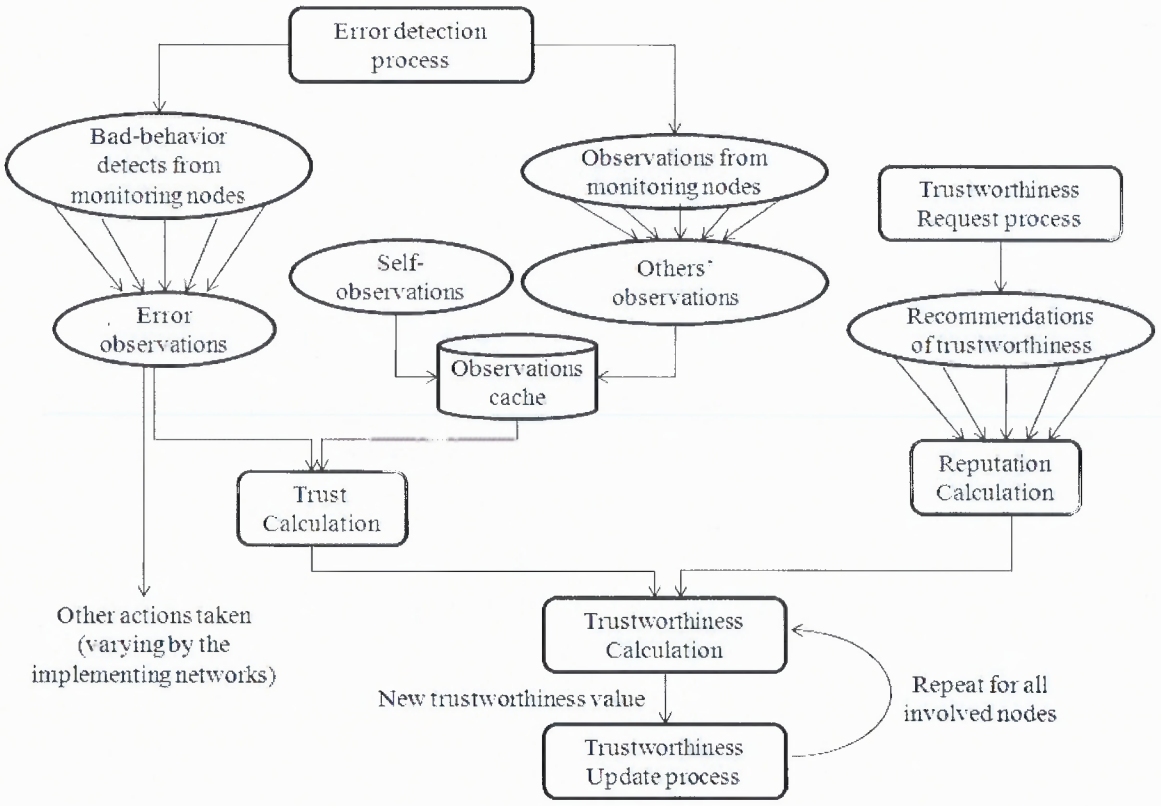


Figure 4.3 An illustration of the trustworthiness reevaluation process.

Once the new possible trustworthiness value of a node of interest is determined, the value is tested by the trustworthiness update process to determine if the new possible value is a result of attacks or any misbehavior. If the process does not find any suspicious change in the trustworthiness value, the new possible trustworthiness value becomes the new trustworthiness value. If any suspicious change is detected, the new possible trustworthiness value is rejected. No update occurs then.

4.3.7 Trustworthiness Update Process

After the re-evaluation process, the node ensures that the change is not resulted from any attacks, such as the defaming or collusion attacks. There are ways to detect attacks as follows:

1. By checking the number of changes (both increasing and decreasing) that the trustworthiness value varied in the past sessions.

- If the number of changes fluctuates widely, that may indicate the node associated with this trustworthiness value is under the intermittent disruptive of service attack in which case that the attack takes place intermittently in a short period of time.
- If the number of changes (decreasing) occurs in a very short period of time, that may indicate the node associated with this trustworthiness value is under a defaming attack in which case the attack tries to rapidly decrease the trustworthiness value of a specified victim.
- If the number of changes (increasing) occurs in a very short period of time, that may indicate the node associated with this trustworthiness value tries to falsely increase its trustworthiness value.

2. By comparing the old values with the new ones of other nodes. If the difference is less than the update threshold (η_{update}), the node can update the trustworthiness value of other nodes; otherwise, it rejects the update.

$$\left| \frac{TW_{Y,X}^{Si} - TW_{Y,X}^{Si-1}}{TW_{Y,X}^{Si-1}} \right| \leq \eta_{update}$$

The trustworthiness update process can be depicted in the following figure 4.4.

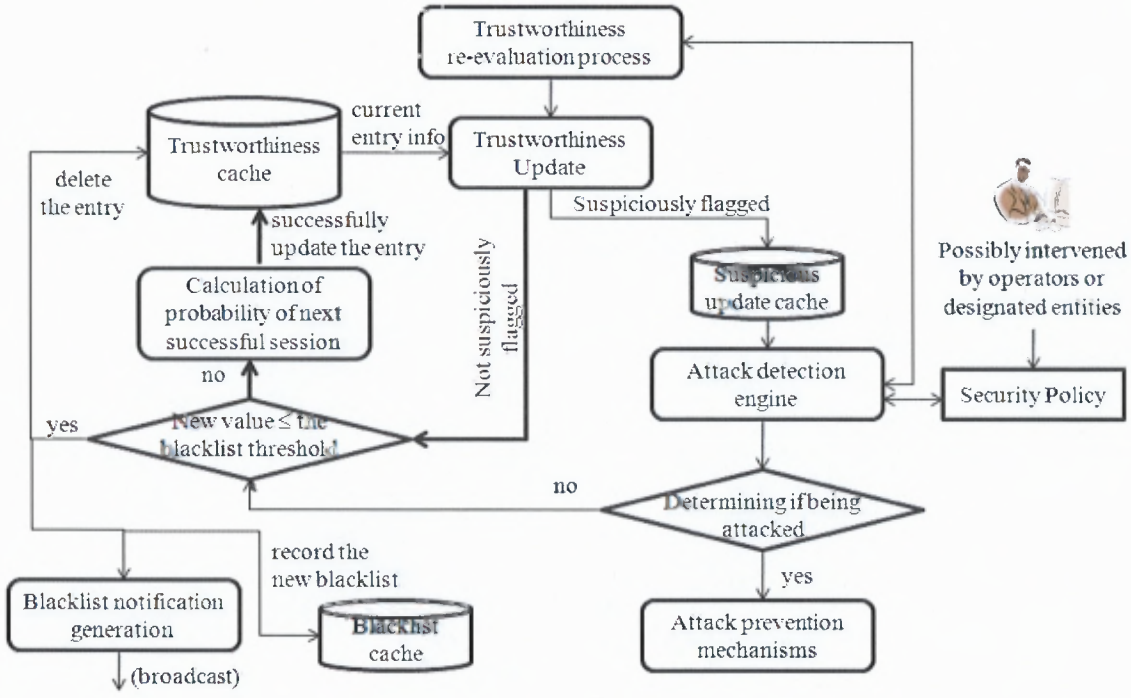


Figure 4.4 An illustration of the trustworthiness update process.

This dissertation suggests that the security policy should be only determined by authorized entities to ensure that the attack detection mechanisms work properly and universally to make the integration of the trust and reputation system more scalable.

4.3.8 Forecasting

There are several protocols that the proposed trust and reputation system can be integrated, such as routing protocols. If the proposed system is integrated into a routing protocol, the system enables several operations as follows: providing crucial information to the routing protocol when selecting intermediate nodes on the forwarding path; providing some probabilistic forecasts on possible behaviors of participating intermediate nodes and outcomes of the transmission; and evaluating behaviors and updating trustworthiness for each intermediate node along the forwarding path that helps forwarding packets and properly follows the given routing protocols.

To forecast the behaviors of participating entities, the proposed system uses the probability model described in Appendix A to calculate some probabilistic forecasts on the future activities, such as the probability of the successful transaction at the next session in which all selected intermediate nodes are on the forwarding path, given past recorded behaviors of those nodes, and the probability that one specified node will provide a true recommendation on trustworthiness of a particular node.

The true recommendation of trustworthiness value of a node is the recommendation given by the node that receives the trustworthiness request message, and that recommendation resides in the majority number of all recommendations, which are proceed through the trustworthiness reevaluation process and are eventually updated through the trustworthiness update process. Otherwise, the false recommendations are referred to two types of recommendations: the recommendation given by the node and later resides in the minority number of all recommendations, or the recommendation given by the node and resides in the majority number of all recommendations but fail to be updated through the trustworthiness reevaluation and update processes.

The node that sends out the trustworthiness request packets keeps tracks of all reply messages. If one or multiple of nodes continuously give out false recommendations on a particular node that eventually lead to an isolation (a node is regarded as untrustworthy) or a quickly increase of trustworthiness of that particular node, it is likely that the system is under an attack from those nodes.

Based on the probability model in Appendix A, the probability that the next transaction is successful in which node X originates and node Y forwards, base on given past D records of $n+p$ sessions, can be presented as $\Pr\left(Tr_{XY}^{Si+1}=1\middle|D\right)$. Therefore, the

probability of the successful transaction at the next session in which all intermediate nodes are on the forwarding path can be calculated from

$$\Pr\left(Tr_{XN_1}^{Si+1}=1|D\right)\times\Pr\left(Tr_{XN_2}^{Si+1}=1|D\right)\times...\times\Pr\left(Tr_{XY}^{Si+1}=1|D\right)\times...\times\Pr\left(Tr_{XN_n}^{Si+1}=1|D\right).$$

Similarly, node X can calculate the probability that the recommendation given by node Y regarding the node of interest Z, based on given past G records of $n+p$ recommendations, can be presented as $\Pr\left(Tr_{XY|Z}^{Si+1}=1|G\right)$, where n is referred to as false recommendations and p as trust recommendations. If the probability is lower than a certain level, node X may not send the trustworthiness request message to node Y and may not include recommendations given by node Y into the trustworthiness reevaluation process.

4.4 Threat Model against the Trust and Reputation System

This section discusses the threat models that aim not only to manipulate the processes and operations of the trust and reputation system, but also to disrupt the availability and credibility of the trust and reputation system. Defense mechanisms are also discussed in this section.

4.4.1 Classifications of Attacks against the Trust and Reputation System

This dissertation explores several attack characteristics of attacking agents. The attacks against trust and reputation systems in communication networks can be categorized in multiple perspectives: motivations, approaches, collaboration, targets.

A Classification based on Motivations

Most trust and reputation systems outline threat models with two major attacks: selfish attack and malicious attack.

A.1 Selfishness

In some networks where the network resources are very limited, the attack with a selfish intention usually does not cause any malicious damages, but instead attempt to gain more advantages over network resources, such as bandwidth. With the selfish intent, the attacker is generally a single entity who does not perform proper actions, such as stop forwarding packets, with respect to performance matters, i.e., saving power resources for its own use. However, the attacker does not disrupt the network activities or not violate the integrity and confidentiality of the transversed packets. Despite, the attacker can perform improper behaviors damaging the credibility of the trust and reputation system that results in a higher trustworthiness value in favor of its benefits for future transactions.

A.2 Malicious Intent

The attacks with any malicious intention can attempt to inflict damages not only to network activities and packets transversed over the networks, but also to the trust and reputation system. One of the objectives of this work is to not directly mitigate some attacks against network activities, but to indirectly monitor such attacks and punish the agents proportionally to the attacks such that every network entity recognizes that there is a consequence to perform improper actions. This indirectly mitigates or reduces the number of attacks against network activities and ensures the necessity of the trust relationship among network entities. However, when the trust and reputation system is

introduced, the system itself can be also the target for attacking agents who want to manipulate the system to gain advantages or even to benefit in helping their attacks to be more effective and inflict larger damages.

B Classification based on Collaboration

This category consists of two groups based on a number of attacking agents.

2.1 Act-alone

There is physically a single attacking agent who plans the attack, collects information before the attacks, and launches the attacks alone. However, it is also likely possible that the physically single agent may create a number of fake entities, submit those fake entities into the network, and manipulate or coordinate those entities to launch the attack. The Sybil attack can be classified into this category by the fact that an agent may inject the network with multiple entities and control them to deliberately launch other types of attacks.

2.2. Collusion

Multiple attacking agents can collude with each other to launch attacks. It is possible that not all attacking agents actively involved in the attacks. Some of these agents may passively collect information regarding network activities or specific victims and relay such information to the other agents who use the information to actively attack the targets.

C Classification based on Targets

The attacks can incur on the trust and reputation system, the information system, and other layer protocols such as routing protocol or anonymous protocol.

4.4.2 Attacks on the Trust and Reputation System

This attack targets either the trustworthiness value or the trust and reputation system.

A Attacks on the Trustworthiness Value

An attacking agent in this kind generally attempts to manipulate or build an influence into the trustworthiness reevaluation process and update process to either increase the trustworthiness value of its target, i.e., itself or its colluded agent, or decrease the trustworthiness value of its target, i.e., a specific victim. There are two attacks that are categorized in this group: self-promoting and defaming. This work discusses the details of the two attacks and investigates scenarios that an attacking agent can do to achieve the anticipated result.

- *Self-promoting attack*

The goal of this attack is to increase the trustworthiness values in two ways: by replying trustworthiness request messages and falsely exaggerating the higher-than-actual trustworthiness value, with or without help from its colluded agents; and by giving falsely satisfactory observation messages to help its colluded agents gain the higher-than-actual trustworthiness values.

- *Defaming (or slander) attack*

The attacking agent attempts to defame victims in two ways: by sending route error messages to falsely accuse victims of performing improper actions such that the victims' trustworthiness values are decreased; and by falsely replying trustworthiness request messages about a victim with a lower-than-actual trustworthiness value. This kind of attack is usually effective when a sufficient number of agents collude with each other

to target one or more victims because the agents become a majority in manipulating the trustworthiness of these victims.

B Attacks on Services of the Trust and Reputation System

This type of attack generally causes a questionable problem on reliability, and availability of services offered by the system. An attacking agent in this kind simply disrupts the services to create an impression of all entities that the system is simply not trustworthy to be relied on. In addition, the agent may try to attack the trustworthiness values of unspecified victims to demonstrate that some or all services, such as trustworthiness reevaluation, update, and request processes, are easy to be deceived and manipulated. Once network entities experienced such a drawback or weakness of the system, the entities simply no longer employ the trust and reputation system.

- *Selective Disruption of Service*

An attacking agent selectively discards trustworthiness request and replies messages that are originated from or destined to a specific victim such that the victim cannot update its trustworthiness cache.

- *Intermittent Disruption of Service*

An attacking agent performs malicious actions for some periods of time, with one or multiple strategies, and once the trustworthiness value starts to decline to the certain value, such as the blacklist threshold, the agent performs proper actions for the other periods of time to regain the reputation value. Once the agent's reputation value rises to be sufficiently high, the agent performs malicious actions again.

- *Low-rate Disruption of Service*

Similar to the intermittent disruption of service, the attacking agents perform malicious actions for a very short period of time, through one or multiple strategies, that target specific victims and proper actions for a long period of time. It is difficult to detect such behavior in distributed networks, such as ad hoc networks, without central authority that consecutively monitor and constantly detect such behavior.

- *Re-entry attack*

The attacking agent performs malicious actions until its trustworthiness value drops below the certain value, i.e., blacklist threshold, and it is completely ignored by the other nodes in the network. Thereafter, the agent changes its identity and re-enters the network to perform malicious actions again.

4.4.3 Attacks on the Information System

This attack focuses on information that is being transmitted throughout the network. There are basically two types of packets: control packets and data packets. Some attacks may attempt to disrupt the protocol or system by blocking control packets, modifying contents in control packets, or injecting false contents or fake control packets. Some other attacks may instead target data packets without disrupting the protocol or system. The contents in the data packets are the main target that the attackers are after. The other attacks may aim at both control and data packets to concurrently achieve multiple goals.

4.4.4 Attacks on Other Protocols or Applications

This type of attack varies depending on the protocol or application into which the trust and reputation system is applied, such as routing protocol or anonymous protocol. The attack exploits weaknesses or vulnerabilities of the implementation of the trust and

reputation system on the other protocol. Therefore, the defense techniques must be developed dependently because one defense technique effectively works well on one applied protocol may not work at all if the same technique is applied onto another protocol.

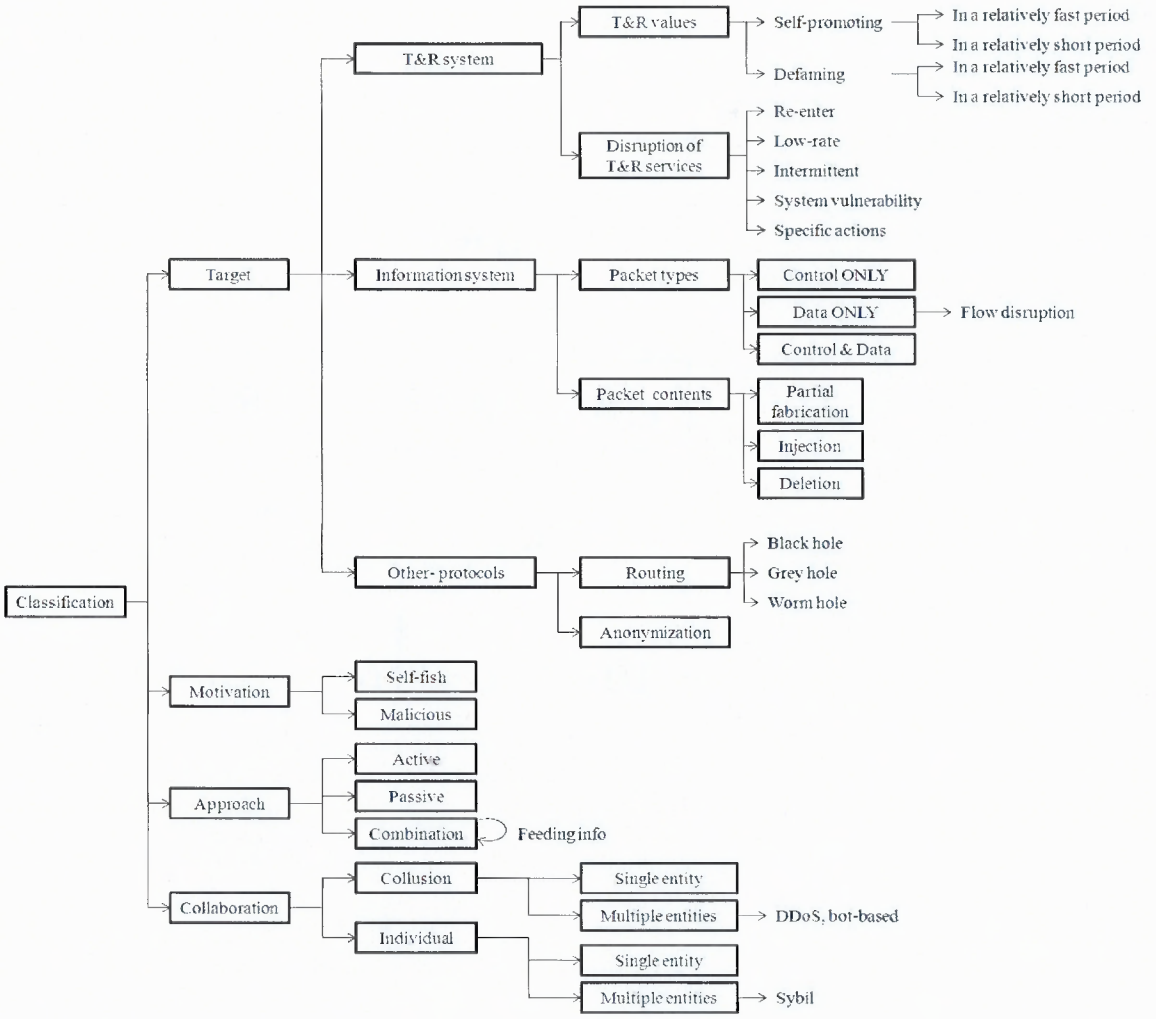


Figure 4.5 A Classification of Attacks.

CHAPTER 5

ELLIPTIC CURVE CRYPTOSYSTEM-BASED GROUP KEY MANAGEMENT FOR SECURE GROUP COMMUNICATIONS

5.1 Objective

Group key management (GKM) prevents all non-membership users from acquiring the group key to decrypt the encrypted messages broadcasted among group members. This chapter expands the utilization of elliptic curve cryptosystem (ECC) into GKM to decrease the key lengths while providing securities at the same level as that of other cryptosystems, and propose the cluster scheme to make the key management more efficient. A group is divided into smaller groups, called clusters, which are independent from each other particularly in the operations of key management. The cluster key is used to secure the selection and distribution of a group key while the group key is used to encrypt broadcast messages. When a membership changes, only the corresponding cluster and its cluster key and a fraction of the group key are changed, thereby reducing the key computation and decreasing time for group key management.

5.2 Introduction

In general, existing GKM protocols can be categorized into three classes [41]-[47]: centralized, decentralized, and distributed. The centralized based GKM protocols perform a key selection and distribution through a central authority, which may potentially be the cause of a point of failure. The distributed based GKM protocols enable all group members to fully establish the group key without a central authority so that every member contributes a corresponding part to the group key, but the members are required to store all related information and to compute the group key themselves. The

decentralized based GKM protocols select temporarily any group member to act as a central authority for key selection and distribution. In this chapter, a hybrid of decentralized and centralized based schemes is introduced to place a central coordinator to keep most of group information, and to establish the group key in a distributed manner.

The public key cryptosystem ensures the security of key distribution for the symmetric key cryptosystem, but the public key cryptosystem itself incurs high computational intensity. Therefore, a mix of public key and symmetric cryptosystems is introduced where the public key cryptosystem is used for key distribution and authentication, and the symmetric key cryptosystem is used for message encryption. ECC is a public key cryptosystem and considered to have several major advantages of smaller key length and more efficient computations over other public key cryptosystems, such as discrete logarithm based cryptosystems (i.e., Diffie-Hellman) or factorization based cryptosystems (i.e., RSA), while it provides the same security level [47]-[56]. Thus, it may be deployed particularly in networks where the devices, such as chips embedded in smart cards, have limited resources such as CPU power, processing time, and storage.

The contributions of this chapter include:

1. This work proposes to utilize the elliptic curve cryptosystem for group key management such that the group key is securely selected, rekeyed, and distributed in group communications with a shorter key length but provides the same security level as that of other cryptosystems.
2. This work proposes the cluster scheme for GKM to decrease the processing time and key computations.

5.3 Background Information

5.3.1 Elliptic Curve

An Elliptic Curve E is defined by two variables, x and y, and a Weierstrass equation as follows:

$$y^3 = x^3 + Ax + B,$$

where A and B are constant coefficients, and a set of A, B, x, y is an element of a field. In this paper, the field is a finite field for a prime p, and the coefficients must satisfy the condition:

$$4A^3 + 27B^2 \neq 0.$$

Below are several properties related to the elliptic curve that can be adapted and used in the elliptic curve cryptosystem, and only those that are deployed in this ECC-GKM scheme are presented. The complete details of elliptic curve and number theory are covered in Reference [48].

4.1 Two Points addition. The group law enables the operations for adding together two points $P=(x_P, y_P)$ and $Q=(x_Q, y_Q)$ on the elliptic curve, resulting in the third point $R=(x_R, y_R)$ on the same elliptic curve, defined as $P +_E Q = R$, where $+_E$ denotes the point addition operation and neither point is a point at infinity (∞). The third point's coordinates can be calculated from Table 5.1.

Table 5.1 The Point Addition Properties [3]

$x_P \neq x_Q$	$x_R = \left(\frac{3x_P + A}{2y_P}\right)^2 - 2x_P,\ y_R = \left(\frac{3x_P + A}{2y_P}\right)(x_P - x_R) - y_P$
$x_P = x_Q, y_P \neq y_Q$ $P = Q,\ y_P = 0$	$P +_E Q = \infty$
$P = Q,\ y_P \neq 0$	$x_R = \left(\frac{y_Q - y_P}{x_Q - x_P}\right)^2 - x_P - x_Q,\ y_R = \left(\frac{y_Q - y_P}{x_Q - x_P}\right)(x_P - x_R) - y_P$

4.2 Existence of inverses. Given the point P, there exists an inverse point P', which is basically a reflection of P across the x-axis, such that $P +_E P' = \infty$. Therefore, the inverse point P' is also denoted as $-P$. For the Weierstrass equation, given $P = (x_P, y_P)$, the inverse point of P is defined as $P' \equiv -P = (x_P, -y_P)$. Thus, $a(P) +_E a(P') \equiv a(P) +_E a(-P) = \infty$.

4.3 Existence of identity. Given the point P, the identity property is determined as $P +_E \infty = P$.

4.4 Endomorphism of curve E. Given two points P_1 and P_2, $\alpha(P_1 +_E P_2) = \alpha(P_1) +_E (P_2)$.

4.5 Associativity. Given any three points P_1, P_2, P_3 on curve E, $(P_1 +_E P_2) +_E P_3 = P_1 +_E (P_2 +_E P_3)$.

4.6 Commutativity. Given any two points P_1, P_2 on curve E, $P_1 +_E P_2 = P_2 +_E P_1$.

4.7 Point multiplying with integer. When a point multiplies with a positive integer, a, the outcome aP can be found by repeatedly adding itself a times, which is very expensive for a large integer. The successive doubling operation can be instead used to improve the

efficiency of multiplication particularly when the integer is very large, as follows: $P +_E P = 2P, 2P +_E 2P = 4P, 4P +_E 4P = 8P, 4P +_E P = 5P$.

4.8 Modulation. All operations are in deed modulated by prime p, but for simplicity, the expression $(mod\ p)$ is not shown in this dissertation.

This chapter proposes the elliptic curve cryptosystem based key management by utilizing the above properties. For examples, it follows from Properties 1, 2, and 7 that

$$\begin{aligned} a(P) +_E a(P') +_E b(P) &\equiv a(P) +_E a(-P) +_E b(P) \\ &= \infty +_E b(P) = b(P) \end{aligned}$$

and from Properties 1, 2, 3, 4, 5, and 7 that

$$\begin{aligned} (P_1 +_E P_2) +_E P_3 +_E P_2' &= (P_1 +_E P_2) +_E P_2' +_E P_3 \\ &= P_1 +_E (P_2 +_E P_2') +_E P_3 = P_1 +_E P_3. \end{aligned}$$

5.3.2 Elliptic Curve Cryptosystem (ECC)

The elliptic curve has been applied in cryptography [49]. The security of ECC relies on the strength of the elliptic curve discrete logarithm problem (ECDLP). ECDLP is to find the secret PIN (positive integer) a, given points Q and P, whereas $Q = a(P)$.

The two users agree on the elliptic curve E and point P. User A randomly generates the secret PIN a multiplied by the point P, thus creating its public key $a(P)$, along with information of curve E and point P. Upon the receipt of A's public key, user B randomly generates the secret PIN b multiplied by the given point P to create its public key bP, and sends the public key back to user A. Consequently, users A and B calculates the group key of which only they know by multiplying their secret PIN a and b with B's public key aP and A's public key as $b(aP) = baP$ and $a(bP) = abP \equiv baP$, respectively,

as illustrated in Figure 5.1. This group key $ab(P)$ is used to encrypt and decrypt the messages transmitted between them.

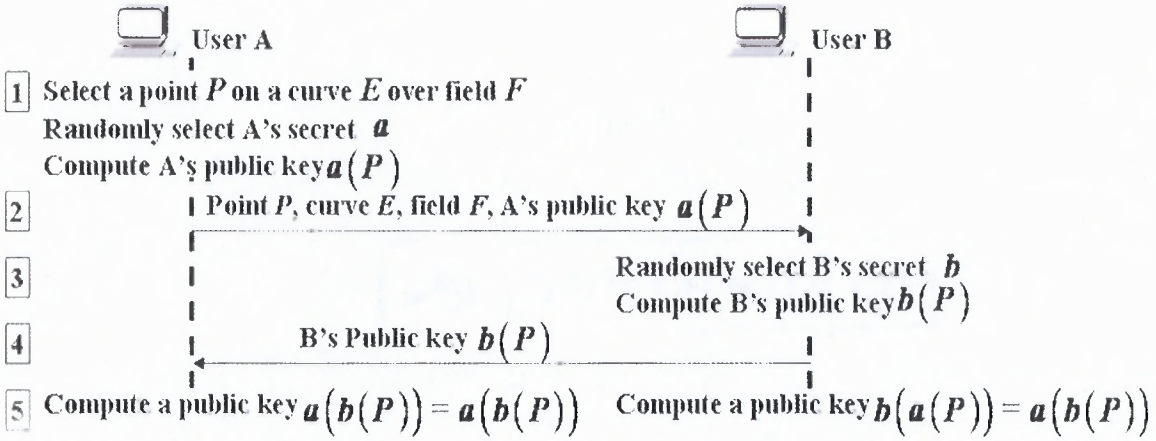


Figure 5.1 An illustration of the ECC-based key exchange [3].

In ECC, a message can be encrypted in two ways: embedding the message into a point, P_M, on curve $E(F_q)$ [50], and bitwise operations such as exclusive-OR (XOR) operation. The embedding of a message simply transforms the message string onto a point of the given elliptic curve.

5.3.3 Key Management Schemes in Group Communications

The key management is a big and fundamental part of many group communications schemes that offer security properties such as access control, authentication, integrity, and so on. A strong key management scheme consists of four major parts: key selection, key distribution, membership management, and rekeying management. The key selection should be executed to provide the strong group key and the key distribution should be carried out through secure channels such that only current members can access the group key. The membership management is also another major part of the key management.

When there is a change in membership, i.e., when one member leaves or one new member joins in, the group key should be rekeyed to assure its secrecy and to ensure two properties: forward secrecy and backward secrecy. The forward secrecy assures that a former member cannot use the old group key to decrypt the messages encrypted with the new group key. The backward secrecy assures that the new group member cannot use the new group key to decrypt the messages encrypted with the old group key, assuming that those messages have been previously intercepted by the new member. This is illustrated in Figure 5.2. The periodic rekeying assures that the group key is secure by periodically changing the group key after a session ends. After the rekeying operation, the two keys should be independent, i.e., the new key is independent of the old key.

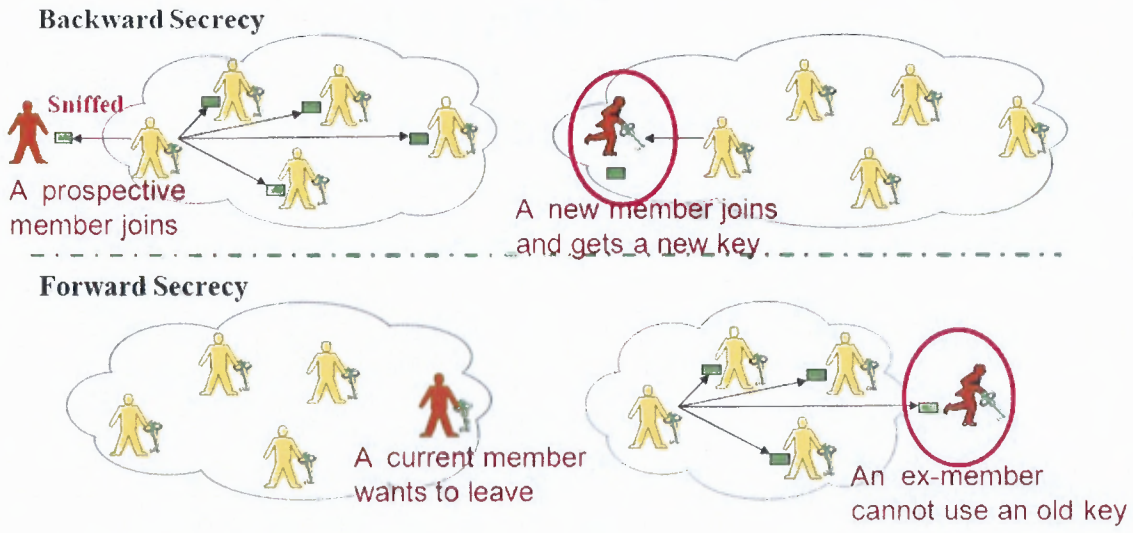


Figure 5.2 An illustration of the key secrecy.

There are several issues on the membership management, including which entity to keep track of the membership status and which entity to respond to the member's malicious behavior, as discussed in Reference [47]. Three major schemes have been proposed in order to handle the membership status: centralized schemes in which the

central authority keeps the whole information; distributed schemes in which each member keeps the whole information; and fully distributed schemes in which each member keeps a part of the whole information. The centralized scheme is suitable for the scenario in which the least information is required to be stored in each member.

5.3.4 Cluster Based Group Key Management

In the ECC-GKM scheme, the group is divided into smaller groups, called clusters. There are two communication levels: group level and cluster level, as illustrated in Figure 5.3. Group messages are broadcasted throughout the group and secured by the group key while cluster messages are broadcasted within the cluster and secured by the cluster key. Let N be the total number of group members, and M be the number of cluster members in each cluster; there will be $\lceil N/M \rceil$ clusters, assuming that each cluster has the same number of members. Let $\{M\}$ and $\{\lceil N/M \rceil\}$ be sets of indices for all cluster members in a cluster and for all cluster heads in a group, respectively.

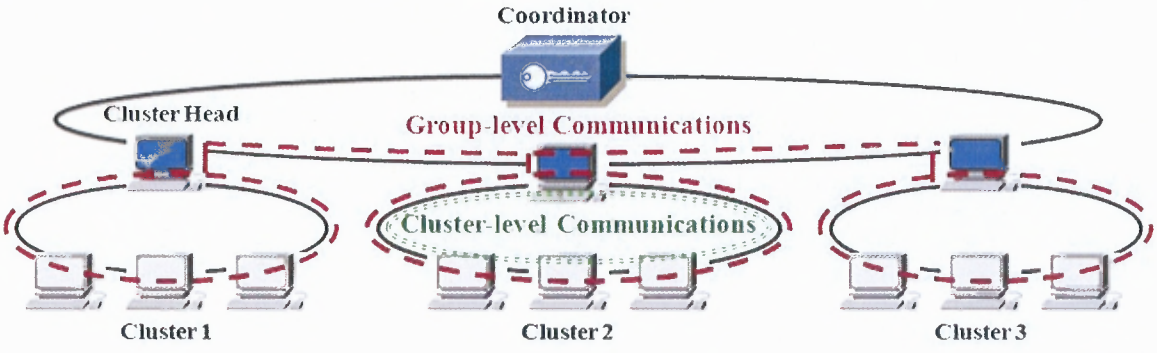


Figure 5.3 Two levels of communications [3].

The cluster head is selected by the coordinator among cluster members to perform the managerial tasks. The cluster head position is held for one session and can be renewed if the coordinator is not reassigned to another cluster member and if the cluster head is still within the same cluster. The coordinator assigns the session period to the cluster head. The coordinator supplies the cluster head with a list of candidate cluster points, and the cluster head picks one of those to be its cluster point so that the cluster point P for each cluster may not necessarily be the same. The coordinator is the authorized entity that performs the following: generates the elliptic curve E, the group point Q, and a set of candidate cluster points; authenticates the new users and assigns them to the clusters; supplies the new members with appropriate information, and communicates with the cluster heads; and stores a list of cluster members and a part of the cluster key for each cluster.

5.4 ECC-based Key Management Scheme

There are two shared keys in the ECC-GKM scheme:

1. The group key is used to encrypt and decrypt the messages broadcasted among the group members.
2. The cluster key is used to encrypt and decrypt the cluster-level messages broadcasted to all cluster members.

Consequently, the key management in the ECC-GKM scheme can be separated into two groups: a cluster key establishment and a group key establishment.

5.4.1 The Cluster Key Establishment

Adapted from a protocol without member serialization in large networks [46], the cluster members are not necessarily numbered serially and the process of generating the cluster key may not be in serial as required in other key management schemes such as group Diffie-Hellman (GDH) protocols. The cluster key can be established within two rounds, as shown in Figure 5.4.

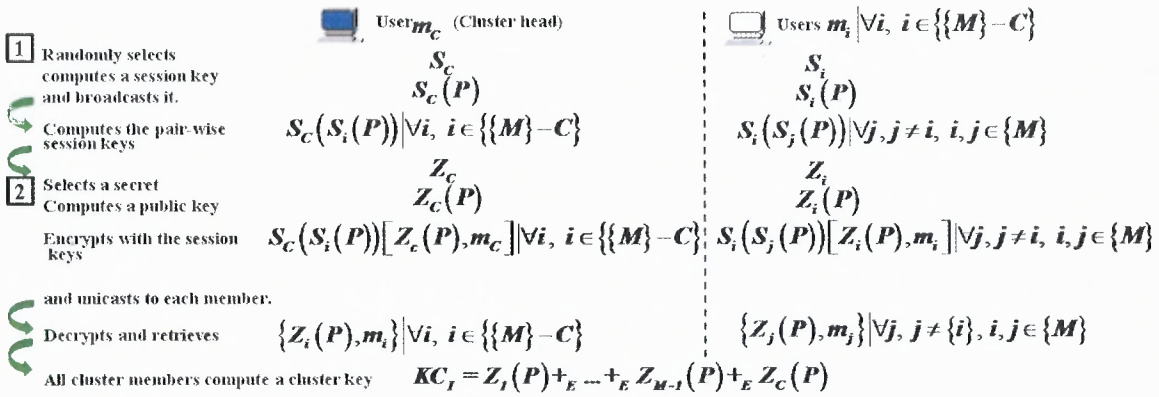


Figure 5.4 A cluster key establishment [3].

In the first round, each cluster member m_i randomly selects a cluster session PIN S_i, computes a cluster session key $S_i(P)$, and broadcasts $\{S_i(P), m_i\}$ to all other cluster members. Upon receipt of the cluster session key from m_j for $\forall j, j \neq i, i, j \in \{M\}$, the pair-wise cluster session key is computed as $S_i(S_j(P))$ for every other cluster member. Note that the cluster head denotes C as its index and $C \in \{M\}$. Thus, the cluster head also computes its cluster session key $S_C(S_i(P))$ for $\forall i, i \in \{\{M\} - C\}$. In the second round, each cluster member m_i selects its cluster secret PIN Z_i, computes its cluster public key $Z_i(P)$, encrypts it with the pair-wise cluster

session key $S_i\left(S_j(P)\right)\left[Z_i(P), m_i\right]$ for $\forall j, j \neq i, i, j \in \{M\}$, and unicasts $S_i\left(S_1(P)\right)\left[Z_i(P), m_i\right]$,..., $S_i\left(S_M(P)\right)\left[Z_i(P), m_i\right]$ to corresponding members. Similarly, the cluster head also computes its cluster public key and encrypts it with the pair-wise cluster session key as $S_C\left(S_i(P)\right)\left[Z_C(P), m_C\right]$ for $\forall i, i \in \{\{M\} - C\}$. Each cluster member including the cluster head decrypts these encrypted cluster public keys by $S_i\left(S_j(P)\right)\left[S_j\left(S_i(P)\right)\left[Z_j(P), m_j\right]\right]$ for $\forall j, j \neq i, i, j \in \{M\}$, retrieves every cluster public key $Z_j(P)$ for $\forall j, j \neq i, i, j \in \{M\}$, and computes a cluster key via some pre-defined function $f\left(Z_j(P) \middle| \forall j\right)$ such as $f\left(Z_j(P) \middle| \forall j\right) = Z_1(P) +_E \ldots +_E Z_M(P)$. The cluster key of the I^{th} cluster is

$$KC_I = Z_1(P) +_E \ldots +_E Z_M(P).$$

Once the cluster key has been established, the cluster head sends a part of the cluster key to the coordinator as

$$KC_I - Z_C(P) \equiv Z_1(P) +_E \ldots +_E Z_{M-1}(P).$$

5.4.2 The Group Key Establishment

After the cluster key is established, the cluster head notifies the coordinator, which then sends the communication halt notification to all cluster heads and subsequently to all group members. The process of the group key establishment is similar to that of the cluster key establishment by comparing the cluster keys as a cluster public key and the cluster header as a cluster member. The hierarchical cluster presented in Reference [46] is partially adapted but the main differences are that there are only two layers, one for the cluster level and another for the group level communications such that it is lightweight,

and the key computations are executed by the group members to avoid the single point of failure in case the key server is attacked. The group key can be established within two rounds.

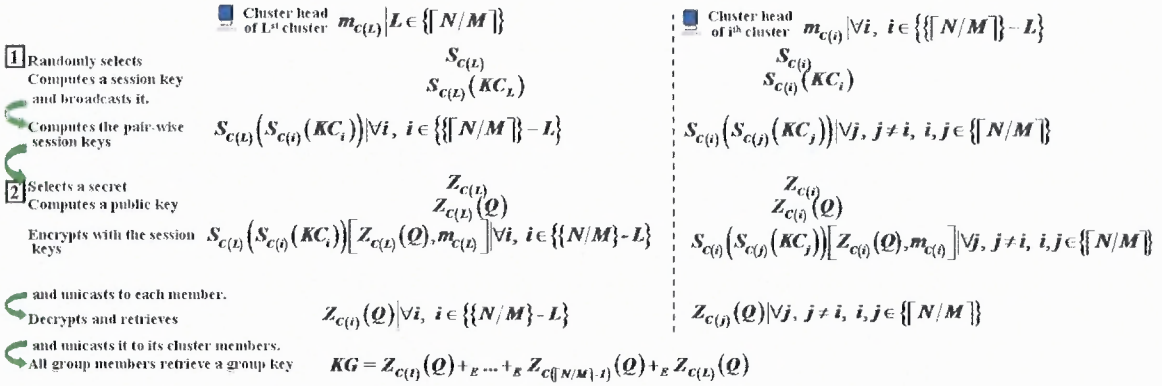


Figure 5.5 A group key establishment [3].

In the first round, each cluster head $m_{C(i)}$ for $\forall i, i \in \{\lceil N/M \rceil\}$ randomly selects a group session PIN $S_{C(i)}$, multiplied by its cluster key $KC_i = (x_{C(i)}, y_{C(i)})$ to generate its group session key $S_{C(i)}(KC_i)$, and then broadcast $\{S_{C(i)}(KC_i), m_{C(i)}\}$ to all other cluster heads $\forall j, j \neq i,\ i, j \in \{\lceil N/M \rceil\}$. Upon receipt of the group session key from other cluster heads $m_{C(j)}$ for $\forall j, j \neq i,\ i, j \in \{\lceil N/M \rceil\}$, the cluster head computes the pair-wise group session key $S_{C(i)}(S_{C(j)}(KC_j))$ for $\forall j, j \neq i,\ j \in \{\lceil N/M \rceil\}$. In the second round, each cluster head selects its group secret PIN $Z_{C(i)}$, computes its group public key $Z_{C(i)}(Q)$, encrypts it with the pair-wise group session keys $S_{C(i)}(S_{C(j)}(KC_j))[S_{C(i)}(Q), m_{C(i)}]$ for

$\forall j, j \neq i,\ j \in \{\lceil N/M \rceil\}$, and unicasts it to corresponding cluster heads. Any cluster head $m_{C(i)}$ decrypts these encrypted group public keys by $S_{C(i)}\left(S_{C(j)}\left(KC_j\right)\right)\left[S_{C(j)}\left(S_{C(i)}\left(KC_i\right)\right)\left[Z_{C(j)}(Q), m_{C(j)}\right]\right]$, retrieves each group public key $Z_{C(j)}(Q)$ for $\forall j, j \neq i,\ i, j \in \{\lceil N/M \rceil\}$, and computes a group key as

$$KG = Z_{C(1)}(Q) +_E \ldots +_E Z_{C(\lceil N/M \rceil)}(Q).$$

Denote $m_{C(L)}$ as the cluster head of cluster L, and Figure 5.5 illustrates the group key establishment operation. Each cluster head encrypts the group key with its cluster key, $KC_i[KG]$ for $\forall i, i \in \{\lceil N/M \rceil\}$, and broadcasts the encrypted group key to its cluster members. The cluster members decrypt it with the cluster key and retrieve the group key.

Once the group and clusters have been established, there can be three incidents: a new user requested to join, a current cluster member requested to leave, and a cluster head requested to leave. These membership-changed rekeying operations are to assure that the forward and backward secrecies are intact, as discussed below:

5.4.3 Individual Join

The new user, m_X, submits the joining request to the coordinator, which sends back information, such as curve E, and points P, Q, in which the new user is referred by the index $X \in \{M\}$. The user who wants to join the group starts the individual joining processes as follows. The user sends the coordinator a request along with its identity. The coordinator authenticates the request whether the user actually sends the request. Note

that this step may be carried out off-line. A coordinator's public key is issued to the new user that will be used to decrypt the messages sent from the coordinator during the new cluster head selection process, which will be discussed shortly. In addition, the new member also issues its pair-wise key to the coordinator that will be used by the coordinator to decrypt the messages sent from this new member in the future. The coordinator also sends the user information, including curve E, point P, field F_P, cluster ID to which the user will be assigned, cluster head ID, and part of the existing cluster key $KC_L - Z_C(P) \equiv Z_1(P) +_E \ldots +_E Z_{M-1}(P)$. Simultaneously, the coordinator notifies the cluster head of the new cluster member to start the cluster key and group key rekeying operations. Subsequently, the cluster head issues a communication halt notification to its existing members. The cluster key rekeying operation for the joining member can be executed within two rounds to guarantee the backward secrecy. Figure 5.6 illustrates the cluster key rekeying operation when the new member joins in.

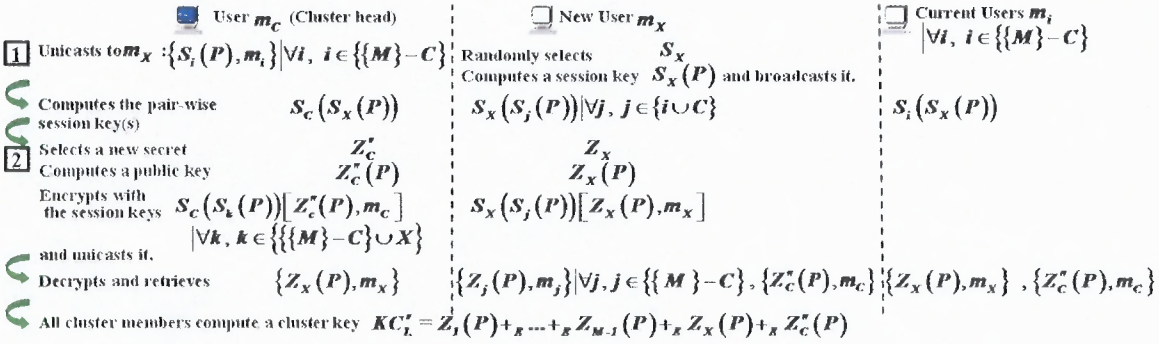


Figure 5.6 A cluster key rekeying operation when the new member joins in.

In the first round, the cluster head unicasts the list of cluster session keys, $\{S_i(P), m_i\}$ for $\forall i, i \in \{\{M\} - C\}$ that are used by other cluster members, to the new member. Concurrently, the new member randomly selects a cluster session PIN S_X, computes a cluster session key $S_X(P)$, and broadcasts $\{S_X(P), m_X\}$ for $\forall j, j \in \{M\}$ to all current cluster members, including the cluster head. In the second round, upon receipt of the m_X's cluster session key, the cluster head and other cluster members compute the pair-wise cluster session key, $S_i(S_X(P))$ for $\forall i, i \in \{M\}$, used between themselves and the new member. Similarly, the new member computes the pair-wise cluster session keys, $S_X(S_j(P))$, for $\forall j, j \in \{M\}$. Then, the cluster head selects its new cluster secret PIN Z_C'', computes its new cluster public key $Z_C''(P)$, encrypts it with every pair-wise cluster session keys $S_C(S_k(P))[Z_C''(P), m_C]$ for $\forall k, k \in \{\{\{M\} - C\} \cup X\}$, which includes every member and the new member, and unicasts them. Similarly, the new member selects its cluster secret PIN Z_X, computes its cluster public key $Z_X(P)$, and encrypts it with every pair-wise cluster session keys $S_X(S_j(P))[Z_X(P), m_X]$ for $\forall j, j \in \{M\}$, and broadcasts them. Existing cluster members obtain $\{Z_C''(P), m_C\}$ in addition to $\{Z_j(P), m_j\} | \forall j, j \in \{\{M\} - C\}$, which is also obtained by the cluster head, while the new member obtains all cluster public keys $\{Z_j(P), m_j\} | \forall j, j \in \{\{M\} - C\}, \{Z_C''(P), m_C\}$. Eventually, the new cluster key can be derived as follows:

$$KC_L'' = (KC_I) +_E Z_C(P') +_E Z_C''(P) +_E Z_X(P)$$
$$= (Z_1(P) +_E \ldots +_E Z_{M-2}(P) +_E Z_C(P)) +_E Z_C(-P) +_E Z_C''(P) +_E Z_X(P)$$
$$= Z_1(P) +_E \ldots +_E Z_{M-2}(P) +_E Z_X(P) +_E Z_C''(P).$$

Note that all cluster members always keep the cluster head's cluster public key in order to obtain $Z_C(P') \equiv Z_C(-P)$, where an inverse point P' is a reflection of P across the x-axis, such that $P +_E P' = \infty$. Therefore, the inverse point P' is also denoted as $-P$. The cluster head updates the coordinator with the new cluster key as $KC_L - Z_C(P) = Z_1(P) +_E \ldots +_E Z_{M-1}(P)$. Therefore, the coordinator keeps the cluster key from which the cluster head's cluster public key is subtracted, rather than the actual cluster key. During this cluster key rekeying operation, the group key is still unchanged and used by all other clusters. Since the cluster key KC_L is changed to KC_L'', the group key must be consequently updated to support the backward secrecy. The cluster head of membership-changed cluster, $m_{C(L)}$, randomly selects a new group session PIN $S''_{C(L)}$, computes a new group session key $S''_{C(L)}\left(KC_L''\right)$, and broadcasts it to other cluster heads. This cluster head $m_{C(L)}$ also selects a new group secret PIN $Z''_{C(L)}$, computes a new group public key $Z''_{C(L)}(Q)$, and encrypts it with a new group session key such that $S''_{C(L)}\left(S_{C(j)}\left(KC_j\right)\right)\left[Z''_{C(L)}(Q), m_{C(L)}\right]$ for $\forall i, i \in \left\{\left\{\lceil N/M \rceil\right\} - L\right\}$, and unicasts them to other corresponding cluster heads. Upon receipt of this new group session key, every other cluster head $m_{C(i)}$ computes a new pair-wise group session key $S_{C(i)}\left(S''_{C(L)}\left(KC_L''\right)\right)$

for $\forall i, i \in \left\{\left\{\lceil N/M \rceil\right\} - L\right\}$. Every other cluster head also decrypts this encrypted new group public key and computes the new group key, derived as

$$\begin{aligned} KG'' &= KG +_E Z_{C(L)}(Q') +_E Z''_{C(L)}(Q) \\ &= KG +_E Z_{C(L)}(-Q) +_E Z''_{C(L)}(Q) \\ &= Z_{C(1)}(Q) +_E \ldots +_E Z_{C(\lceil N/M \rceil - 1)}(Q) +_E \ldots +_E Z''_{C(L)}(Q). \end{aligned}$$

Every cluster head updates the new group key to their cluster members. Figure 5.7 illustrates the group key rekeying operation when the new member joins in.

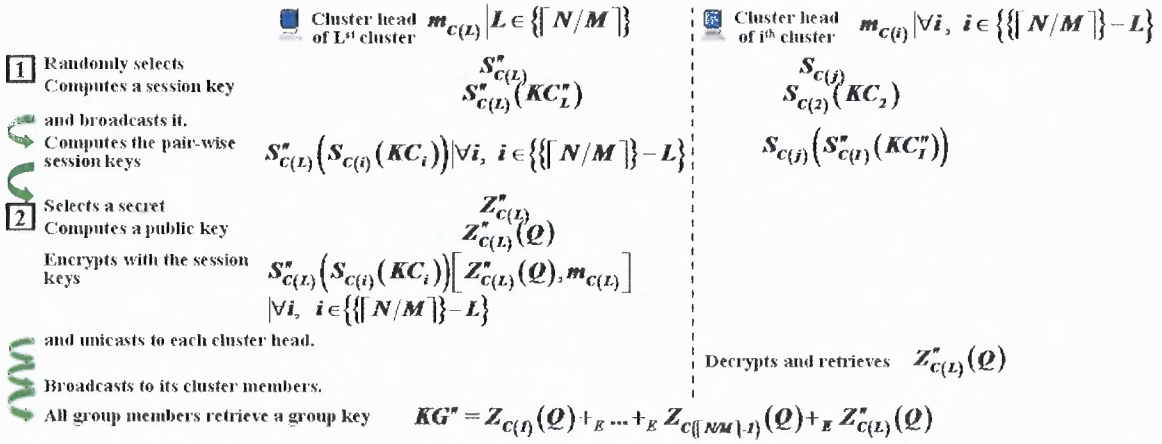


Figure 5.7 A group key rekeying operation when the new member joins in.

5.4.4 Individual Departure

When the cluster member leaves the group, the leaving member should be considered whether it is the cluster member or the cluster head. To ensure the forward secrecy, the cluster key used in the cluster to which the leaving member belongs must be changed so that the leaving member cannot decrypt future messages broadcasted within the cluster, and the group key must be changed so that the leaving member cannot decrypt future messages encrypted with the current group key. When the cluster head detects any

change in the individual membership, the cluster head issues the communication halt notification to all cluster members, and starts to establish the new cluster key. In addition, the cluster head reports this membership change to the coordinator, which sends the group key change notification to all other cluster heads. Subsequently, other cluster heads issue the communication halt notification to their cluster members along with the temporary group key, which is defined as

$$\begin{aligned}\overline{KG} &= KG +_E Z_{C(L)}(Q')\{\ \} \\ &= Z_{C(1)}(Q) +_E \ldots +_E Z_{C(\lceil N/M \rceil - 1)}(Q) +_E \ldots +_E Z_{C(L)}(Q) +_E Z_{C(L)}(-Q) \\ &= Z_{C(1)}(Q) +_E \ldots +_E Z_{C(\lceil N/M \rceil - 1)}(Q).\end{aligned}$$

Therefore, communications among other clusters are still active by using this temporary group key until the new group key is rekeyed. Since only the cluster that the membership changed is affected, the overall performance of group communications is improved.

The cluster key rekeying operation for the leaving member can be executed in the same way the cluster key is established. The main difference is that the leaving member is not included. The new cluster key can be derived as

$$\begin{aligned}KC_L''' - Z_C'''(P) &= Z_1'''(P) +_E \ldots +_E Z_{M-1}'''(P) +_E \left(Z_C'''(P) +_E Z_C'''(P')\right) \\ &= Z_1'''(P) +_E \ldots +_E Z_{M-1}'''(P) \\ KC_L'' &= Z_1(P) +_E \ldots +_E Z_{M-1}(P) +_E Z_X(P) +_E Z_C''(P).\end{aligned}$$

The cluster head randomly selects a random secret PIN $R_{C(L)}$, computes $R_{C(L)}(P)$, and reports the membership change and part of the new cluster key, $\left\{R_{C(L)}(P), m_Y, KC_L''' - Z_C'''(P)\right\}$, to the coordinator. This report is encrypted with the coordinator's public key so that only the coordinator with the session PIN key can

decrypt the message. The coordinator updates the list of cluster members for this cluster and sends an acknowledgement back to the cluster head, encrypted with $R_{C(L)}(P)\left[\{m_Y\}\right]$. The group key rekeying operation for the leaving member is derived in the same manner of the group key rekeying operation for the joining member, which yields the new group key as

$$\begin{aligned} KG''' &= \overline{KG} +_E Z'''_{C(L)}(Q) \\ &= Z_{C(1)}(Q) +_E \ldots +_E Z_{C(\lceil N/M \rceil - 1)}(Q) +_E Z'''_{C(L)}(Q). \end{aligned}$$

5.4.5 Cluster Head Departure

When the cluster head is leaving, it must notify the coordinator so that the coordinator begins a new cluster head selection process. The cluster head randomly selects a random secret PIN $R_{C(L)}$, computes $R_{C(L)}(P)$, and encrypts $\{R_{C(L)}(P), m_{C(L)}\}$ with the coordinator's public key. The cluster head also sends the communications halt notification to all cluster members. The coordinator may select the new cluster head by the trust and reputation approach, which is beyond the scope of this dissertation. After the coordinator selects the new cluster head, the selection notification, encrypted with the coordinator's session PIN key and signed by the coordinator, is multicast to the selected member and the leaving cluster head. The leaving cluster head verifies whether it is authentic and then makes the transition notification broadcasted to all cluster members along with the new cluster head identification. The rekeying operations begin to obtain the new cluster key and the group key.

5.4.6 Periodic Rekeying

The cluster key is changed to safeguard its secrecy regardless of the membership status. Note that the session period for each cluster may not be the same because the periodic rekeying operation is executed concurrently for all clusters. If the coordinator does not select the new cluster head and the old cluster head is still in the cluster, the old cluster head automatically inherits the cluster head position. The periodic rekeying operation starts when the coordinator broadcasts the periodic rekeying notification to all cluster heads, which then broadcast the communication halt notification to their cluster members. The cluster key and group key rekeying processes are similar to those of the cluster key and group key establishment operations that were previously discussed.

5.5 ECC-based GKM Analysis

The performance key establishment operation and rekeying operation of the ECE-GKM scheme is summarized in Table 5.2, in terms of the following parameters as in Reference [45]: the number of rounds, the number of messages sent to and received by any member, and the number of "*point multiplication with integer*" operations. Note that, in Table 5.2, $\varnothing$ means null, and m_S can be either the new joining member m_X or the leaving member m_Y. In the ECC-GKM scheme, the key independence property is true because only portions of group keys contributed by the membership-changed clusters are replaced. The ECC-GKM scheme offers both forward and backward secrecy in key establishment and rekeying operations. Unlike other cryptosystems such as the GDH scheme, ECC-GKM does not require a member serialization in the two operations such that none of the members needs to wait for each other to configure the keys, but time synchronization

among the members is required to respond promptly to the membership changed. The memory used by each member is reduced from the group level to the cluster level, thereby reducing the memory requirement, as shown in Table 5.2.

Table 5.2 The Performance of Key Establishment in the ECC-GKM Protocol [3]

Properties		Establishment operation	
		Cluster Level	Group Level
The number of rounds		2	2
Overhead bound		$O(M^2)$	$O\left(\left(\lceil N/M \rceil\right)^2\right)$
The number of messages sent by m_i, $m_{C(j)}$		M, $M+1$	0, $\lceil N/M \rceil + 2$
The number of messages received by m_i, $m_{C(j)}$		$2(M-1)$, $2(M)-1$	1, $2\left(\lceil N/M \rceil\right)-1$
The number of point multiplication with integer operations by m_i, $m_{C(j)}$		$M+1$, $M+1$	0, $\lceil N/M \rceil + 1$
Keys Storage	coordinator	Pair-wise keys with all group members / A part of cluster keys The coordinator's public key and private key	
	cluster head	Pair-wise key with the coordinator / The coordinator's public key All cluster member's public keys / Cluster key All cluster head's public keys / Group key	
	member	Pair-wise key with the coordinator The coordinator's public key / Cluster key / Group key	
Properties		Rekeying operation	
		Cluster Level	Group Level
The number of rounds		2	2
Overhead bound		$O(M^2)$	$O\left(\left(\lceil N/M \rceil\right)^2\right)$
The number of messages sent by m_i, $m_{C(j)}$, m_s		0, $2(M+1)$, M	0, $\lceil N/M \rceil + 2$, $\varnothing$
The number of messages received by m_i, $m_{C(j)}$, m_s		3, $(M+1)$, 2	1, $2\left(\lceil N/M \rceil\right)-1$, $\varnothing$
The number of point multiplying with integer operations by m_i, $m_{C(j)}$, m_s		1, $(M+1)$, $(M+1)$	0, $\lceil N/M \rceil + 1$, $\varnothing$

5.6 Conclusion

In this chapter, ECC is incorporated into GKM to decrease the key length while providing the same security level as that of other cryptosystems, and adapt the cluster based key management scheme to make the ECC-GKM scheme more efficient. A group is separated into several clusters, which are independent from each other, particularly in the operations of key management. The cluster key is used to secure the group key selection and distribution while the group key is used to encrypt broadcast messages. When there is a membership change, the corresponding cluster key and the group key are changed in order to protect their secrecies. The periodic rekeying operation also strengthens the key secrecies.

CHAPTER 6

ADAPTIVE TRUST-BASED ANONYMOUS NETWORK

6.1 Objective

A novel adaptive trust-based anonymous network (ATAN) is proposed. The distributed and decentralized network management in ATAN does not require a central authority so that ATAN alleviates the problem of a single point of failure. In some existing anonymous networks, packets are routed onto intermediate nodes anonymously without knowing whether these nodes are trustworthy. On the other hand, an intermediate node should ensure that packets which it forwards are not malicious, and it will not be allegedly accused of involving in the attack. To meet these objectives, the intermediate node only forwards packets received from the "trusted" predecessor, which can be either the source or another intermediate node.

In ATAN, the proposed trust and reputation model aims to enhance anonymity by establishing a trust and reputation relationship between the source and the forwarding members. The trust and reputation relationship of any two nodes is adaptive to new information learned by these two nodes or recommended from other trust nodes. Therefore, packets are anonymously routed from the "trusted" source to the destination through "trusted" intermediate nodes, thereby improving anonymity of communications.

6.2 Introduction

In recent years, security and privacy issues in communication networks receive considerable attention and many schemes have been proposed to achieve several objectives. One objective is to make the communications anonymous with several

properties: source anonymity, destination anonymity, and unlinkability between the source and destination. Anonymous networks can be managed in a centralized or distributed manner [57]-[73]. Many centralized anonymous networks, such as PENET [58], have been targeted by the attacker who seeks to obtain the identification of the sender and receiver nodes and other pertinent information, and by authorities who want to force the network administrator to shut down the network services or to turn over users' information. In this chapter, a framework of the adaptive trust-based anonymous network (ATAN) is proposed to provide anonymous communications based on the proposed trust and reputation model.

The rest of this chapter is organized as follows: Section 6.3 gives a theoretical background of some previous research on anonymous network protocols and discrete logarithm problem-based cryptosystems. Section 6.4 presents the framework of ATAN. Section 6.5 discusses the threat model. Section 6.6 offers a brief network analysis, followed by the conclusion and future work in Section 6.7.

6.3 Background Information

A brief overview on anonymous network protocols as well as the cryptosystems based on the discrete logarithm problem is discussed.

6.3.1 Anonymous Networks

The path routing approach in anonymous protocols can be classified into fixed-length path or variable-length path approaches. The fixed-length path approach requires the sender node to know *a priori* the network topology such that intermediate nodes along the path can be chosen before the sender's packet is transmitted. This approach also

needs to determine parameters for path selection such as the number of hops in order to fix the path length. The major drawbacks of this approach include the requirement of the knowledge of the whole network topology for path selection, and the large overhead required for packet encryption for all intermediate nodes. The variable-length path approach defines the path length as a random variable that uses the probabilistic algorithm to determine whether the packet is forwarded among the members or sent directly to the destination node. Its major drawback is the intensive computation.

Some anonymous network protocols have proposed various concepts of the trust and reputation model in the peer-to-peer networks [62]-[66].

6.3.2 Diffie-Hellman Cryptosystem

Diffie and Hellman [74] proposed the Diffie-Hellman (DH) cryptosystem to securely exchange keys between two or more users, and its security strength depends on the discrete logarithm (DL) problem, as illustrated in Figure 6.1.

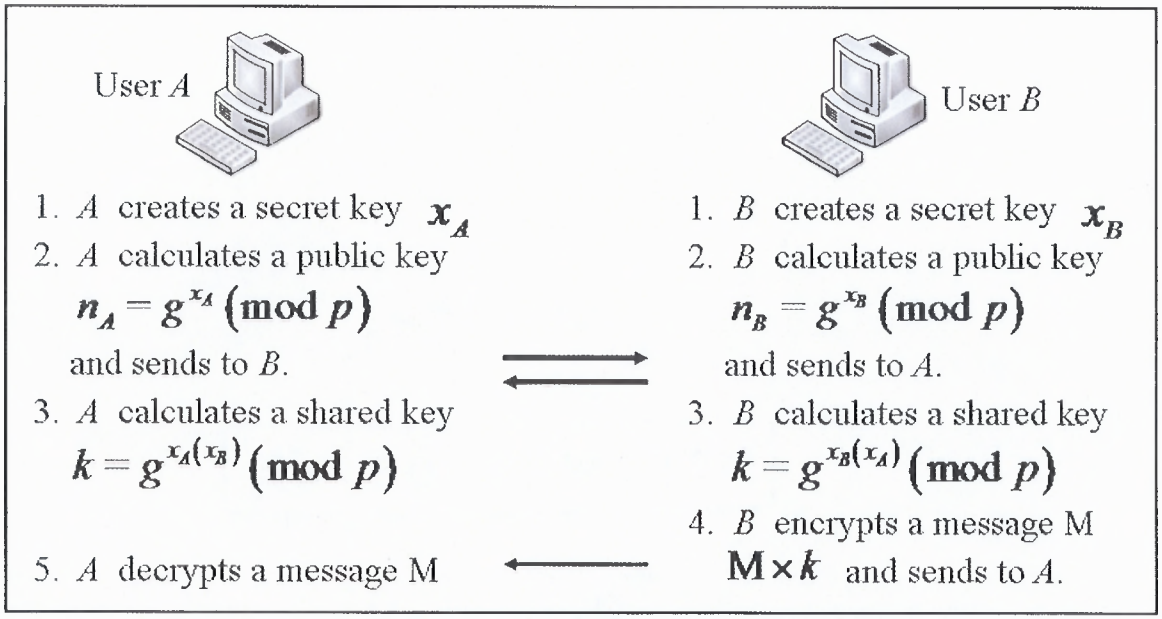


Figure 6.1 An illustration of the Diffie-Hellman cryptosystem [4].

The cryptosystem has two system parameters: p, a large prime number selected from the finite field F_p, and g, an integer number less than p. The discrete logarithm problem is to compute the shared key K or the secret keys X_A and X_B, given the public keys k_{X_A} and k_{X_B}. The DH cryptosystem is also utilized in group communications [47], [61], [74]-[76]. ElGamal [75] proposed the ElGamal public key cryptosystem by converting the Diffie-Hellman cryptosystem into a public key cryptosystem as illustrated in Figure 6.2.

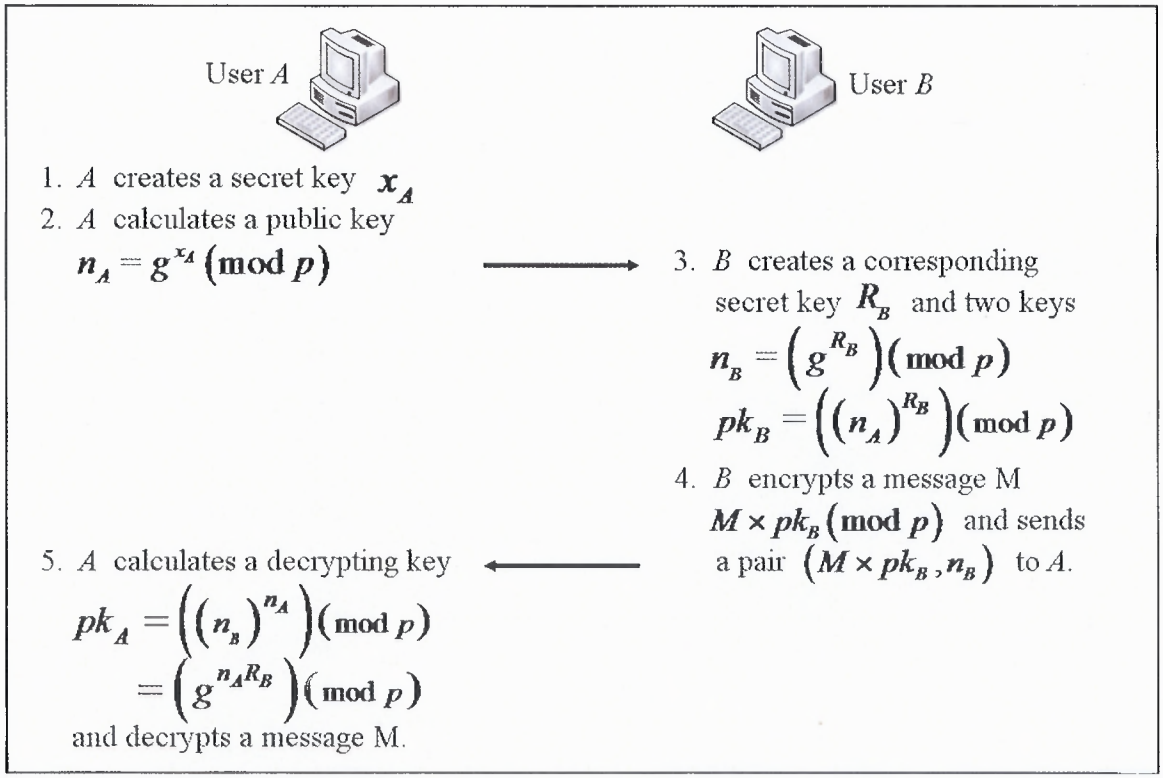


Figure 6.2 An illustration of the ElGamal cryptosystem.

6.4 ATAN Framework

The ATA (adaptive trust-based anonymous) protocol is an application-layer protocol, but sits below other application-layer protocols. From a network point of view, a network that deploys the ATA protocol is called an ATA network (ATAN). ATAN sits on top of the existing core networks and has a virtual topology connecting all members together. From the network management point of view, ATAN can be viewed as a loosely decentralized network because all members are grouped into clusters, each managed by a cluster head. There is no central authority that can become a potential single point of failure. The cluster head is changed periodically to tighten the security and to share the administrative responsibility among members.

ATAN uses any underlying routing protocols to route the packets from one intermediate node to another through the physical path (one node represents a router) in the core network, but the path established by ATAN is basically a logical path above the physical path (one node represents a member). Therefore, the terms "node" and "member" will be used interchangeably throughout this chapter.

This work also extends the concept of group key management and proposes the trust and reputation system and the voting protocol for anonymous networks such that anonymity is insured [47], [62].

Several major advantages of ATAN include:

1. ATAN utilizes the trust and reputation model to enhance anonymity because the packets are routed among "trusted" members in decentralized networks where there is no central authority to control and detect malicious members.

2. ATAN is scalable by dividing the network into smaller clusters based on a trust and reputation value such that the users join and leave ATAN without creating a big impact on the rest of the network while maintaining high anonymity.
3. The large number of members increases anonymity by hiding packets among the members and camouflaging packets among other unspecified packets. The proposed trust and reputation system encourages the well-behaved users to stay longer and to help forwarding other members' packets, rather than to send their packets and then leave, because they will have in return higher anonymity for their own packets.
4. ATAN uses a voting protocol to share the responsibility of managing the cluster among cluster members to defend against a malicious insider attack.

6.4.1 Terminologies, Definitions, Expressions, and Assumptions

To illustrate the concept of ATAN, some details including terminologies, definitions, expressions and assumptions, are first outlined.

6.4.1.(a) Terminologies

The following terminologies are defined.

$\mathbb{Z}$	A set of all existing members in each cluster, $\mathbb{Z} = \{1,...,N\}$.
N_{MAX}, N_{MIN}	The maximum and minimum number of members in each cluster, respectively.
$\mathbb{R}$	A set of all existing clusters in ATAN, $\mathbb{R} = \{1,...,M\}$
M_{MAX}	The maximum number of clusters in ATAN.
$n(i,j)$	A representation of the i^{th} node in the j^{th} cluster, $i \in \mathbb{Z}, j \in \mathbb{R}$, called node ID. Note that every cluster head always has $i=1$, also represented as $n(C,j)$.
$C(n)$	A representation of the cluster to which node n belongs.
γ_A^B	The average trust and reputation (ATR) value of node B known to node A.

$\mathbb{Z}$	A set of all existing members in each cluster, $\mathbb{Z} = \{1,...,N\}$.
$\eta_{C(n)}, \eta_j$	The ATR threshold of the cluster to which node n belongs, and that of the j^{th} cluster, respectively.
$\gamma^{C(n)}, \gamma^j$	The average ATR value of the cluster to which node n belongs, and that of the j^{th} cluster, respectively.
X_n, k_{X_n}	A private key and a public key of node n, respectively.
$\mathbb{U}$	The set of all entries in the ATR database of each member,. $\mathbf{U} = \{(1,1),...,(N,M)\}$
t_j	The initial time when the cluster head of the j^{th} cluster is selected.
τ	The constant lifecycle for every cluster.
$t(i,j)$	The timestamp of node $n(i,j)$'s ATR value kept in the ATR database.
t_Q	The timeout for the query packet.
B	The maximum number of digests in the probing packet.
$H(.)$	The hash function.
$\hat{\mathrm{Y}} \overset{\text{change}}{\Rightarrow} \mathrm{Y}$	The old value of Y is replaced by its new value $\hat{\mathrm{Y}}$.

6.4.1.(b) Definitions

The source node is defined as the end node that generates and sends out packets, and the destination node as the end node to which packets are ultimately destined. The sender node may not generate packets but sends out packets to the other end on the link. The receiver node receives packets from the other end on the link but may or may not be the ultimate destination.

6.4.1.(c) Expressions

To simplify the process proposed throughout the chapter, each process is expressed in five parts: command name, a pair of sender node and receiver node, a descriptive set of the pair, condition or reason, and packet type, defined as:

$$\textit{COMMAND\_NAME}\left[(\textit{SENDER, RECEIVER})|\textit{NODE\_SET} : \textit{CONDITION / REASON}\right](\textit{PACKET\_TYPE})$$

The expression is interpreted as the receiver node performs a command on the packet received from the sender node, and any sender-receiver pair applicable to the command is described in the node_set part.

The command name indicates how to deal with the packet, of which there are three commands in ATAN: verify, accept, and reject commands. In addition, there are four types of user packets: probing (M_P), forwarding data (M_F), replying data (M_R), and error packets (M_E). For each error packet type, the information pattern to be recorded is shown in Table 6.1.

TABLE 6.1 INFORMATION PATTERN FOR ERROR PACKETS [4]

Error #	Information Pattern	Description
1	$[\text{INFO}] = \left[1, Seq, sender\_ID, \gamma_{receiver}^{sender}, receiver\_ID, \gamma_{receiver}^{receiver}, \gamma_{receiver}^{C(receiver)}, \overline{\gamma}_{C(receiver)}\right]$	Bad sender
2	$[\text{INFO}] = \left[2, Seq, sender\_ID, \gamma_{receiver}^{sender}, receiver\_ID, \gamma_{receiver}^{receiver}\right]$	Repeated sequence number, same source
3	$[\text{INFO}] = \left[3, Seq, sender\_ID, \gamma_{receiver}^{sender}, receiver\_ID, \gamma_{receiver}^{receiver}, no\_hop\right]$	No next hop

6.4.1.(d) Assumptions

The following assumptions are made.

1. The destination node is assumed to understand the ATA protocol in order to send back the replying packets anonymously to the source node, but it may not necessarily be an ATAN member.

2. Before the data is inserted into the payload, a filter module must filter out anything that can reveal the sender node's identity. The purpose is to ensure anonymous communications in anonymous connections, as discussed in Reference [61]. However, this filtering is beyond the scope of this chapter since it vastly depends on the application-layer protocol. Assume that such a filter has been executed earlier.

6.4.2 Cluster-based ATAN Network Management

All members in ATAN are divided into clusters, and each cluster is managed by the cluster head. A cluster has an average value of the ATR values of all cluster members (called an average ATR value) and a minimum ATR threshold to indicate the proper range of the ATR value of each member that can stay in the cluster. Therefore, every one in a cluster examines and controls each other such that the average ATR value of the cluster remains unchanged or even improves. For the first-time user, a new user must join the cluster that has the lowest average ATR value, by using the join procedure. When a user receives a sufficiently high ATR value after having performed good-behaved activities at least for a certain time period, that user may join another cluster that has the higher average ATR value, by using the upgrade procedure; this is referred to as a membership upgrade.

Two major benefits of such an upgrade are: first, user A can have a higher cluster threshold $\eta_{C(A)}$ to forward only messages originated by a user, say B, who holds the ATR value satisfying $\gamma_A^B \geq \eta_{C(A)}$; second, the packet originated by user A can be set a higher Trust requirement T such that the intermediate node C, which meets the condition $\gamma_C^D \geq T$, can forward the packet to the next hop D.

For each cluster, the cluster key is used to encrypt all administrative packets, but not data packets. These cluster-level administrative packets include the local query packet, and the packet used in the voting protocol.

For the whole network, the group key is used to encrypt all administrative packets transmitted among cluster heads. These group-level administrative packets are used when the cluster heads update and exchange information about their clusters. This is illustrated in Figure 6.3.

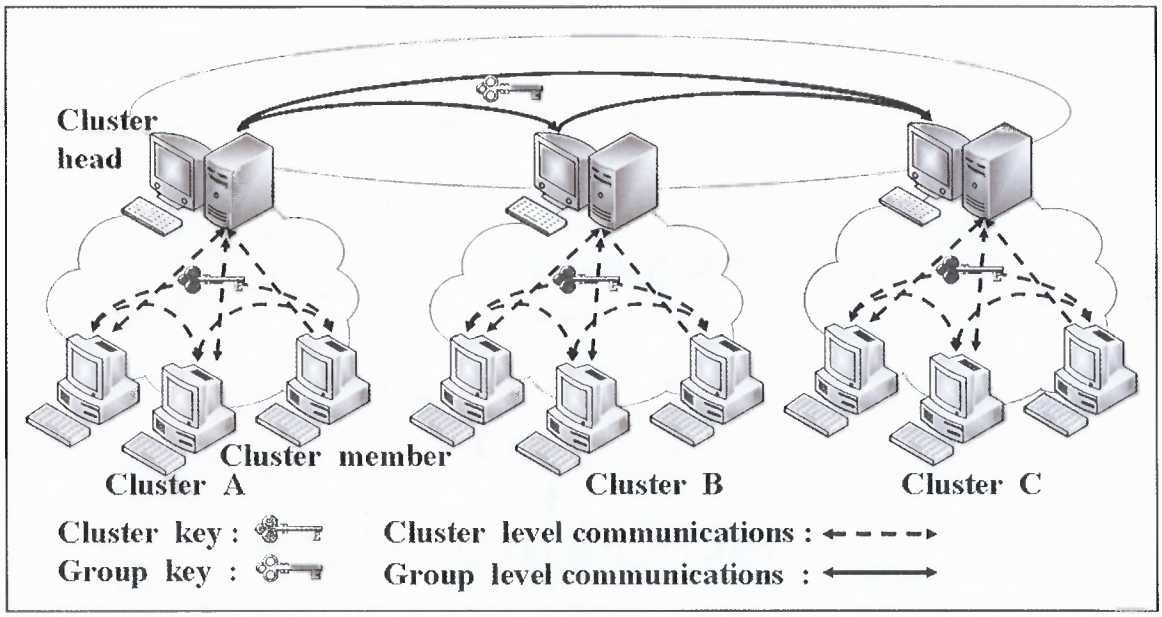


Figure 6.3. An illustration of communication levels in ATAN [4].

There are two main protocols to perform network management in ATAN: clustering protocol and voting protocol.

6.4.2.(1) Clustering Protocol

The clustering protocol is to establish and manage the clusters with three procedures: setup, merge, partition, and to manage the memberships with four procedures: join, leave, upgrade, and downgrade.

1) Setup Procedure

The first cluster in ATAN is formed by the setup procedure as follows.

(*Setup.1*) A user announces itself as an interim cluster head to form the cluster. It selects the parameters for the Diffie-Hellman cryptosystem (such as g, p, and F_P) and creates a one-time public key R_{n_C} and a one-time public key

$$pk_{n_C} = g^{R_{nC}} \pmod{p}.$$

(*Setup.2*) These DH cryptosystem parameters along with information about the sponsor, such as name, IP address, and email address, are signed with its one-time public key $[INFO] \times pk_{n_C}$. The sponsor broadcasts this signed information to other users as an invitation, denoted as

$$INVITATION\left[pk_{n_C} \times [INFO]\right].$$

(*Setup.3*) Each user in a group that wants to form the cluster creates a long-term private key X_{n_i} and a long-term public key

$$k_{n_i} = g^{X_{n_i}} \pmod{p} \text{ for } \forall i.$$

Each user broadcasts its public key.

(*Setup.4*) The interim cluster head receives the public keys from all users, authenticates each user, and creates a long-term private key X_{n_C} and a long-term public key

$$k_{n_C} = g^{X_{n_C}} \pmod{p}.$$

Note that although the user authentication is made before the user can join ATAN, the authentication mechanism is beyond the scope of this dissertation. Some challenge handshake authentication (CHAP) protocols such as Diffie Hellmand-based CHAP may be applied here. Then, the cluster head creates the cluster key as

$$CK = \left(g^{\sum X_{n(i,j)}} \right)^{X_{n(1,j)}} \mid \forall i, i \in \mathbb{Z} \setminus \{1\}, N < N_{MAX}, j = J.$$

Since the interim cluster head's private key is kept secret, only the cluster head can create the cluster key. The cluster head computes the encrypting key corresponding to each user's public key (based on ElGamal cryptosystem) as

$$ek_{n_C}^{n_i} = \left(k_{n_i}\right)^{X_{n_C}} \pmod{p} = \left(g^{X_{n_i}}\right)^{X_{n_C}} \pmod{p} \text{ for every } i.$$

The cluster key and information about the cluster is encrypted with this key $ek_{n_C}^{n_i}$ and unicasts the message to all users.

(*Setup.5*) Each user decrypts the message with the decrypting key

$$dk_{n_i}^{n_C} = \left(k_{n_C}\right)^{X_{n_i}} \pmod{p} = \left(g^{X_{n_C}}\right)^{X_{n_i}} \pmod{p}.$$

For two or more clusters, the group header, say *n(1, L)*, is selected among the cluster heads in a round robin manner. The group header only

manages the group key to which all cluster heads contribute the share (the public key). The group is set up based on the setup procedure. The group key is computed as

$$GK = \left(g^{\sum X_{n(1,j)}} \right)^{X_{n(1,L)}} | \forall j, j \in \mathbb{R} \setminus \{L\}.$$

2) Join Procedure

The join procedure is described below and illustrated in Figure 6.4. Some challenge handshake authentication (CHAP) protocols such as Diffie Hellman-based CHAP can be used to authenticate users.

(*Join.1*) A new user, says node ***n***, sends the request to one or several cluster heads that have the lowest average ATR value of cluster.

(*Join.2*) The cluster head $\boldsymbol{n_C}$ checks whether it can accept the request. If the number of cluster members is less than the maximum ($N < N_{MAX}$), the request is preliminarily accepted; otherwise, rejected. Then, the cluster head sends the invitation back to the user:

$$INVITATION[REQ\_ID, g, p, field\ F_P].$$

If there are multiple invitations, the user may reply to the first invitation that is received or wait for a certain period to collect the invitations and determines to join which cluster and discards the others.

(*Join.3*) Once the cluster is chosen, the new member , *n(N+1, J)*, randomly selects its *private key* and computes a *public key* $X_{n(N+1,J)}$ denoted as

$$k_{n(N+1,J)} = \left(g^{X_{n(N+1,J)}} \right) (\mathrm{mod}\ p),$$

and sends the request along with the public key to the chosen cluster head $n_{C(J)}$.

(*Join.4.1*) The cluster head authenticates the user (i.e., Diffie Hellman-Challenge Handshake Authentication Protocol (DH-CHAP)).

(*Join.4.2*) When authentication is successful, the cluster head broadcasts the suspension of the reciprocation of administrative packets (but not data packets) to all cluster members and computes the new cluster key as follows.

Let $CK_J = \left(g^{\sum Xn_{(i,J)}} \right)^{X_{n(1,J)}} (\text{mod } p) \Big| \forall i, i \in \mathbb{Z}$ be the current cluster key before the new member, *n(N+1, J)*, joins; the cluster head replaces its old *private key* $X_{n(1,j)}$ with the new key $\hat{X}_{n(1,j)}$, and computes the new cluster key

$$\widehat{CK_J} = \left(g^{\sum X_{n(i,J)}} . g^{X_{n(N+1,J)}} \right) \hat{X}_{n(1,J)} (\text{mod } p)$$

$$= \left(g^{\sum X_{n(i,J)}} \right)^{\hat{X}_{n(1,J)}} (\text{mod } p) \Big| \forall i, i \in \{2, ..., N+1\}.$$

(*Join.4.3*) The cluster head creates the encrypting key corresponding to the user's public key:

$$ek_{n_C}^{n(N+1,J)} = (k_n)^{X_{nC}} = \left(g^{X_n R_C} \right) (\text{mod } p).$$

(*Join.4.4*) The cluster head encrypts preliminary information about the current cluster and the signed certificate with $ek_{n_C}^{n(N+1,J)}$ as

$$ek_{n_C}^{n_{(N+1,J)}} \times \Big[INVITATION\_ID, NODE\_ID\big(n(N+1,j)\big), \gamma_{n_C}^{n(N+1,j)}, \Big[ATR\_DATABASE\big(cluster\ j\big) \Big], CERT_{n(N+1,j)} \Big]$$

(*Join.4.5*) The key $k_{n(N+1,J)}$ is stored in the cluster head's *public key database* as the user *n(N+1, J)*'s public key. The associated user information (*NODE_ID*, $\gamma_{n_C}^{n(N+1,J)}$, timestamp) is stored in the cluster head's ATR database.

(*Join.4.6*) The cluster head broadcasts the new cluster key, encrypted with the old cluster key so that only the existing members, not the new member, can decrypt and retrieve the new cluster key.

(*Join.5*) The new member computes the decrypting key $dk_{n_{(N+1,J)}}^{n_C} = \left(k_{n(N+1,J)}\right)^{X_{n_C}} = g^{X_{n(N+1,J)}X_{n_C}} \pmod p$ to retrieve such information.

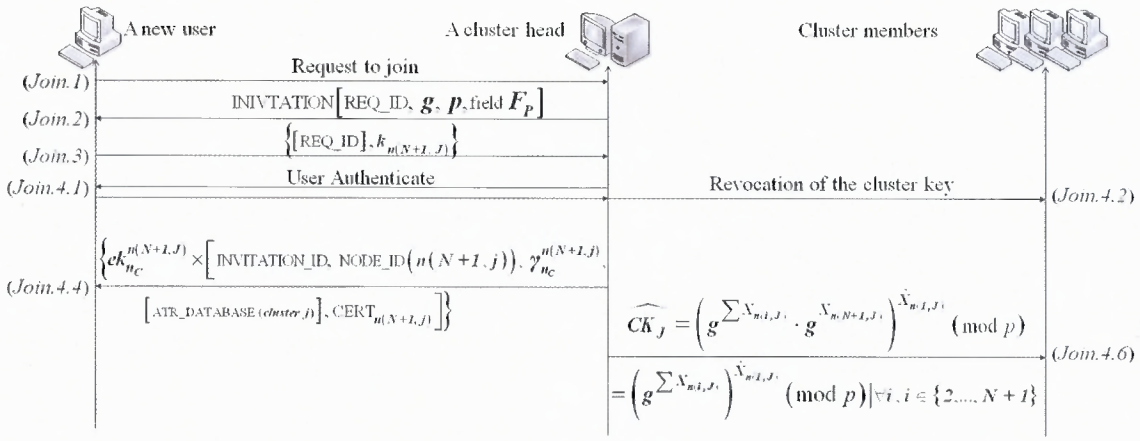


Figure 6.4 An illustration of the JOIN procedure [4].

3) Leave Procedure

There are three possible cases in the leave procedure:

3.1 An existing user, says node *n(N, J)*, leaves ATAN. The cluster head periodically detects the status of all cluster members every t_U (an update period). The cluster head temporarily suspends the reciprocation of all administrative packets (but not data packets) and computes the new cluster key as follows.

(a) Let $CK = \left(g^{\sum X_{n(i,J)}} \right)^{X_{n(1,J)}} \pmod p$ for $\forall i, i \in \mathbb{Z} \setminus \{1\}$ be the current cluster key; the cluster head computes a new cluster key by subtracting the leaving node *n(N, J)*'s public key from the current cluster key, and replacing its old private key $X_{n(1,j)}$ with the new key $\hat{X}_{n(1,j)}$ as

$$CK_j = \left(g^{\sum X_{n(i,J)} - \left(X_{n(N,j)} \right)} \right)^{\hat{X}_{n(1,j)}} \pmod p \text{ for } \forall i, i \in \mathbb{Z} \setminus \{1\}.$$

(b) The new key is encrypted with the public key of all existing cluster members, $ek_{n_C}^{n_i}$, that is stored in the cluster head's public key database.

(c) The cluster head n_C unicasts the encrypted new cluster key, $\left\{ [CK_j] \times \left[ek_{n_C}^{n_i} \right] \right\}$, to every cluster member.

3.2 The cluster head of cluster *j* leaves. The cluster members of this cluster *j* try to contact their cluster head three times after losing contact with the cluster head. If all contacts fail, any node (among those having the highest ATR value in the cluster and staying in the cluster longer than any member), say *n(2, J)*), becomes an interim cluster head. Every member deletes the old cluster head's public key and profile off the public key and ATR databases. The interim

cluster head creates a new pair of the public key and private key ($\hat{X}_{n(2,J)}$ and $\hat{k}_{\hat{X}_{n(2,J)}}$) and computes a temporary cluster key as

$$Temp\_CK_J = \left(g^{\sum X_{n(i,J)} - \left(X_{n(2,J)}\right)} \right)^{\hat{X}_{n(2,J)}} \pmod{p} \text{ for } \forall i, i \in \mathbb{Z} \setminus \{1\}.$$

Then, the interim cluster head calls for vote to select the new cluster head. The ballot is encrypted with the temporary cluster key.

3.3 A time period of the cluster head position expires; the cluster head notifies other members and calls for vote to select the new cluster head. All cluster members vote to select the new cluster head based upon the ATR value of all candidates known to each cluster member.

- *Threat*: A malicious interim cluster head may try to let some colluded members to join. The solution is to check the list of all existing members during the voting for selecting a new cluster head.

Once the new cluster head is selected, it broadcasts its public key to other cluster heads. The group header computes the new group key as

$$\widehat{GK} = \left(g^{\sum X_{n(1,j)} - X_{n(1,J)}} \right)^{\hat{X}_{n(1,L)}} | \forall j, j \in \mathbb{R} \setminus \{L\},$$

and broadcasts it to existing cluster heads, encrypted with each cluster head's public key.

4) Merge Procedure

When two clusters try to merge together, there are three requirements that decide the vote: the average ATR values of two clusters are in close range (i.e., ϕ % difference); the sum of the number of members in two clusters does not exceed the maximum allowed; and the number of members in any cluster that have not yet stayed for at least Δ lifecycle is less than ε %. Note that ϕ, ε, and Δ are decided by the implementation.

The cluster heads of two clusters call for vote to merge. For the last two requirements, corresponding information of two clusters are exchanged to help deciding the vote. If the vote for merging fails, the cluster heads look for other equivalent clusters; otherwise, announce the closure of the cluster. In this case, all cluster members start the join procedure as if it is their first time.

If the vote is passed, two cluster heads integrate the ATR database entries; the cluster head of the cluster that has a higher average ATR value becomes the new cluster head of the merged cluster. This cluster head updates other cluster heads of this merge operation.

5) Partition Procedure

When the number of members in a cluster reaches the maximum, the members vote to select the would-be cluster head for the new cluster by using the voting for a cluster head selection. Once the would-be cluster head is selected, the cluster head divides the cluster J into two smaller clusters (J and L) with an equal number of cluster members. The would-be cluster head becomes the cluster head of cluster L. The two cluster heads update other cluster heads of this partition operation.

6) Upgrade Procedure

Before the user can upgrade the cluster, the user is required to meet some requirements: staying for at least Δ lifecycle in cluster J (to prevent a malicious member from upgrading too fast), holding the ATR value that is greater or equal to the average ATR value of cluster L, and the number of cluster members in cluster L does not exceed the maximum.

A cluster member of cluster J with the ATR value greater than or equal to the average ATR value of cluster L ($\gamma_{n(I,J)}^{n(I,J)} \geq \gamma^{C(L)} \geq \gamma^{C(J)}$) follows the upgrade procedure which is similar to the Join procedure except:

- In Step (*Join.2*), the user must provide the certificate issued by the cluster head of cluster J in which the Start_Time, the beginning time that the user joins cluster J, is recorded. The cluster head of cluster L calls for the vote to determine whether or not the request is granted. The voting result binds to the decision from the cluster head of cluster L.
- In Step (*Join.3*), there can be multiple invitations from multiple clusters that hold

$$\gamma_{n(I,J)}^{n(I,J)} \geq \gamma^{C(L)}, \gamma^{C(H)} \geq \gamma^{C(J)},$$

where $\gamma^{C(L)}$ is in the same level as $\gamma^{C(H)}$. In addition, the user must leave cluster J before Step (*Join.4*) begins.

The difference between the join and upgrade procedures is that the user in the join procedure requests to join ATAN for the first time.

7) Downgrade Procedure

If a member behaves maliciously until the ATR value drops below the minimum ATR threshold (which will be discussed later), the cluster head calls for a vote. If the vote is passed, the cluster head revokes its certificate and deletes its entry in corresponding databases, and broadcasts to all cluster members as well as other cluster heads.

6.4.2.(2) Voting Protocol

The voting protocol in ATAN aims to enhance the clustering protocol's ability to resist threats.

- *Threat*: If a cluster head has too much administrative power to control the cluster, an ill-behaved cluster member will try to behave well until it becomes the cluster head and then starts to inject malicious activities. The voting protocol can reduce such a threat by sharing the responsibility of the cluster head among all cluster members.

The voting protocol is conducted as follows:

(*Vote.1*) The node who calls for the vote distributes the ballot indicating the reason for the vote, requirements and corresponding information, and the result.

(*Vote.2*) A member casts the vote:

(a) to select a new cluster head, a voting member looks into its ATR database and selects the member with the highest ATR value. The result includes a certificate of the voter, node ID, and its associated ATR value.

(b) to merge the cluster, a voting member looks into its ATR database to determine if the average ATR value requirement is met, and to check information obtained from the other cluster heads if the last two

requirements are met. The result includes a certificate of the voter and a bit of 1 (yes) or 0 (no).

(c) to grant an upgrade, a voting member looks into its ATR database to determine with respect to knowledge of the requesting node if this node should be accepted to join. The result includes a certificate of the voter, node ID and the number of recorded lifecycles of that member, and its associated ATR value.

(d) to repel a member, a voting member looks into its ATR database to determine if that member should leave the cluster. The result includes a certificate of the voter, and node ID and associated ATR value of that member.

(*Vote.3*) The result is sent to the sponsor, encrypted with a cluster key.

(*Vote.4*) The sponsor decrypts the encrypted vote, authenticates all voters, collects the results from authenticated voters, and broadcasts the final result.

- *Threat*: If a sponsor node tries to alter the final result, any member can dispute the vote. The sponsor must reveal necessary identification of every voter and its associated result.

Note that the join procedure does not call for vote because there is only one criterion: the number of members is less than the maximum.

6.4.3 Trust and Reputation System

In ATAN, an adaptive trust and reputation system is designed to evaluate the trust and reputation values of every member such that the system encourages nodes to help forwarding the packet, in return for an improvement of its ATR value known to other

members. The packet generated from a node with a high ATR value is more desirable to be forwarded by other nodes. The system is also designed to quickly force the abusing members to retreat from the network by using an additive increment and multiplicative decrement strategy. That is, the node will gradually gain a higher ATR value in an additive manner when forwarding packets properly, but will sharply lose the ATR value in a multiplicative manner when forwarding packets maliciously.

To implement the trust and reputation system, each member $n(i, j)$ for $\forall i \mid i \in \mathbb{Z}, \forall j \mid j \in \mathbb{R}$ stores the ATR database as illustrated in Table 6.2.

Table 6.2 The Exemplary ATR Database of Node $n(1,1)$ [4]

NODE	TYPE	ATR	Average ATR of cluster	Timestamp	Lifecycle	Period of Remaining Time
$n(1,1)$	Cluster head	$\gamma_{n(1,1)}^{n(1,1)}$	$\gamma_{n(1,1)}^{1}$	$t_{(1,1)}$	$\Delta_{n(1,1)}$	$t_1 - \tau$
$n(2,1)$	Cluster member	$\gamma_{n(1,1)}^{n(2,1)}$	$\gamma_{n(1,1)}^{1}$	$t_{(2,1)}$	$\Delta_{n(2,1)}$	$t_1 - \tau$
$n(N,M)$	member	$\gamma_{n(1,1)}^{n(N,M)}$	$\gamma_{n(1,1)}^{M}$	$t_{(N,M)}$	$\Delta_{n(N,M)}$	$t_M - \tau$

U

There are six fields in the ATR database: node ID, type of node, ATR value, average ATR value of the cluster, timestamp, lifecycle, and period of remaining time. The type of node indicates whether this node is in the same cluster (cluster member or cluster head) or out of the cluster (member). The timestamp $t(i, j)$ indicates the last time when the node $n(i, j)$'s ATR value has been updated. The number of lifecycle indicates

how long the node remains operating in the cluster. The period of the remaining time $t_j - \tau$ indicates the remaining time of cluster j before the new cluster head is selected.

The trust and reputation system consists of three processes: ATR query, ATR update, and ATR evaluation. The ATR query process queries the ATR value. The ATR evaluation process evaluates the ATR information obtained from the ATR query process as well as from the probing and data forwarding processes. The ATR update process updates the ATR database.

6.4.3.(1) ATR Query Process

When the path is being evaluated and the source node $n(I, J)$ does not have the ATR value of any forwarding intermediate node $n(K, L)$ or the ATR value of the node is expired, the source flags the node so the query process can be initiated immediately (processing sequence number 5 in Figure 6.5) or wait until the data forwarding process ends (processing sequence number 11 in Figure 6.5), depending on the delay sensitivity of the application.

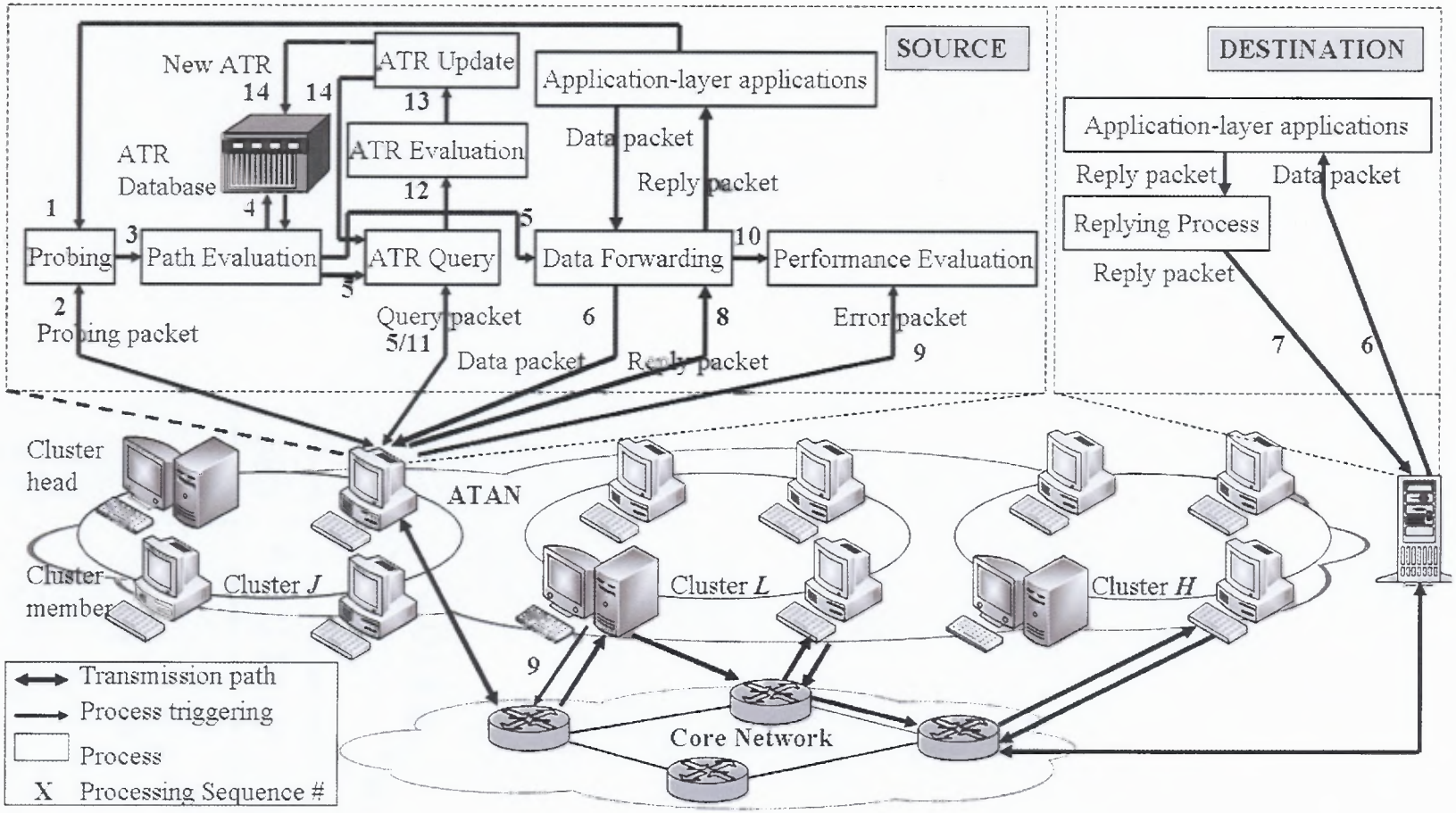


Figure 6.5 An illustration of various processes for data transmission in ATAN [4].

There are two types of ATR query processes: local ATR query (LQ) and global ATR query (GQ).

(a) Local ATR query. In case that node $n(I, J)$ wants to query the ATR value of the target node $n(K,L)\,|\,K \neq I, L \neq J$, node $n(I, J)$ broadcasts the local query about cluster L to other cluster members. All cluster members in cluster J, $n(i,J)\,|\,\forall i, i \in \mathbb{Z}, L \neq J$, examines the ATR value of node $n(K, L)$ known to themselves, $\gamma_{n(i,J)}^{n(K,L)}\,|\,\forall i, i \in \mathbb{Z}, i \neq I$, and replies to node $n(I, J)$.

- *Threat*: The local ATR query can be intercepted and altered. The solution is to encrypt the local query with the cluster key to protect it from an adversary who is not a cluster member. If the local ATR query is falsely given by a cluster member, this is referred to as an insider attack, which will be discussed in Section 6.8.

(b) Global ATR query. In case that node $n(I, J)$ wants to obtain the ATR value of any node outside the cluster, $n(K, L)$, the global query is sent to the cluster head of cluster L, $n(C, L)$. The cluster member may send the query to the other members (but not the target node) in cluster L as well.

- *Threat*: The malicious insider (any ATAN member) can learn which cluster is being sought and alter the result. To prevent this insider attack, the query is encrypted by the querying node with the one-time-use *query key*.

The querying node and the cluster head perform the six following steps:

(*Query.1*) The querying node randomly selects the one-time *query secret key* ($Q_{n(I, J)}$), and computes the query key

$$k_{n(I,J)}^{QUERY\_ID} = \left(g^{Q_{n(I,J)}}\right)(\text{mod } p).$$

(*Query.2*) The querying node sends the query key along with the query information as

$$\left\{\left[\text{QUERY\_ID, CLUSTER\_ID, CERT}_{n(I,J)}\right], k_{n(I,J)}^{QUERY\_ID}\right\}.$$

(*Query.3*) The cluster head authenticates the querying node from its certificate since the public key of the cluster head of cluster J is known.

(*Query.4*) If the querying node is successfully authenticated, the cluster head replies to the querying node with the ATR value of all cluster members without learning which cluster member is sought, randomly selects the secret key corresponding to the query key, ($R_{n(I,L)}$), and computes two keys based on the ELGamal cryptosystem:

$$ek_{n(1,L)}^{QUERY\_ID} = \left(g^{R_{n(1,L)}}\right)(\text{mod } p) \text{ and}$$

$$pk_{n(1,L)}^{QUERY\_ID} = \left(\left(k_{n(I,J)}^{QUERY\_ID}\right)^{R_{n(1,L)}}\right)(\text{mod } p).$$

(*Query.5*) The cluster head sends the replied information encrypted with the key $pk_{n(1,L)}^{QUERY\_ID}$,

$$\left(\left(pk_{n(1,L)}^{QUERY\_ID} \times \left[\gamma_{n(1,L)}^{n(i,L)} \mid \forall i, i \in \mathbb{M}\right]\right), ek_{n(I,J)}^{QUERY\_ID}\right).$$

(*Query.6*) The querying node computes the decrypting key

$$dk_{n(I,J)}^{QUERY\_ID} = \left(ek_{n(1,L)}^{QUERY\_ID}\right)^{Q_{n(I,J)}} (\mathrm{mod}\ p)$$

to retrieve the ATR information $\left[\gamma_{n(1,L)}^{n(i,L)} \middle| \forall i, i \in \mathbf{M}\right]$.

- *Threat*: A malicious cluster head maliciously assigns a high ATR value to some cluster members, particularly if those nodes are its collaborators. If the increase is not steep, the layered trust hierarchy bounds the ATR value of all cluster members to be closely such that this threat is negligible. If the increase is significant, the different threshold strategy used to evaluate the ATR value, which will be discussed shortly, rejects such an increase.
- *Threat*: An adversary can impersonate and modify the reply. The problem can be alleviated by enforcing the cluster head to sign this reply with its certificate such that the replying message is protected and the cluster head cannot repudiate its message.

6.4.3.(2) ATR Evaluation Process

It requires three information processes - querying, probing, and data forwarding - to conduct the ATR evaluation.

From the querying process, the mean of ATR values of $n(K, L)$, obtained from all queried nodes, will be considered as the reputation value of $n(K, L)$ known to the querying node $n(I, J)$ because this ATR value is experienced by other nodes, not by node $n(I, J)$.

From the probing and forwarding processes, the mean of ATR values of the forwarding node $n(K, L)$, retrieved from all digests written by all forwarding intermediate

nodes, will be considered as the trust value of $n(K, L)$ known to $n(I, J)$ because this ATR value is learned by node $n(I, J)$.

Let $^{\mathbb{Q}}=\{(1,1),\ldots,(N,M)\}$ be the set of queried nodes which respond to the ATR query from both local query and global query packets, and $_{\mathrm{B}}=\{(1,1),\ldots,(N,M)\}$ be the set of forwarding nodes which mark their ATR values in each digest.

Upon receiving the query packet, the source, node *n(I, J)*, initiates the ATR evaluation process as follows:

(*Query_Eval.1*) For both local query and global query, node *n(I, J)* computes the ATR value of $n(K, L)$ from a queried node $n(a, b)$ as

$$\gamma_{n(a,b)}^{n(K,L)} \times \gamma_{n(I,J)}^{n(a,b)}.$$

(*Query_Eval.2*) Node *n(I, J)* computes the weighted mean of the ATR value of $n(K, L)$ from all queried nodes of the same cluster b:

$$\gamma''^{\,b}_{n(I,J)} = \sum_{(a,b)}^{\mathbb{Q}} \gamma_{n(a,b)}^{n(K,L)} \times \gamma_{n(I,J)}^{n(a,b)} \Big/ \sum_{(a,b)}^{\mathrm{U}} \gamma_{n(I,J)}^{n(a,b)},$$

where $b \in \mathbb{Z}$.

(*Query_Eval.3*) To further improve the reputation value, node *n(I, J)* multiplies this $\gamma''^{\,b}_{n(I,J)}$ with the average ATR value of cluster b obtained from *n(I, J)*'s ATR database, $\gamma^{b}_{n(I,J)}$:

$$\gamma''^{\,b}_{n(I,J)} \times \gamma^{b}_{n(I,J)}.$$

(*Query_Eval.4*) The outputs from all clusters are summed, and the weighted mean is calculated as the reputation value of node $n(K, L)$ known to node $n(I, J)$, $\mu_{n(I,J)}^{n(K,L)}$:

$$\sum^{\mathbb{Q}} \gamma''^{\,b}_{n(I,J)} \times \gamma^{b}_{n(I,J)} \Big/ \sum^{\mathrm{U}} \gamma^{b}_{n(I,J)} \ ,$$

where $\{I,k\} \in \mathbb{Z}$, $\{J,L\} \in \mathbb{R}$, $J \neq L$.

Similarly, upon receiving the probing and data forwarding packets, the source, node *n(I, J)*, initiates the ATR evaluation process as follows:

(*Prob_Eval.1*) Node *n(I, J)* computes the ATR value of $n(K, L)$ from the forwarding node $n(c, d)$, where $(c, d) \in {}^{\mathrm{B}}$:

$$\sum^{\mathrm{B}} \gamma^{n(K,L)}_{n(c,d)} \times \gamma^{n(c,d)}_{n(I,J)} .$$

(*Prob_Eval.2*) The outputs from all forwarding nodes n(c, d) of the same cluster are summed, and node *n(I, J)* computes the weighted mean:

$$\gamma'''^{\,d}_{n(I,J)} = \sum^{\mathrm{B}} \gamma^{n(K,L)}_{n(c,d)} \times \gamma^{n(c,d)}_{n(I,J)} \Big/ \sum^{\mathrm{U}} \gamma^{n(K,L)}_{n(c,d)} \times \gamma^{n(c,d)}_{n(I,J)} .$$

(*Prob_Eval.3*) To further improve the trust value, node *n(I, J)* multiplies this $\gamma'''^{\,d}_{n(I,J)}$ with the average ATR value of cluster *b* obtained from the *n(I, J)*'s ATR database, $\gamma^{d}_{n(I,J)}$:

$$\gamma'''^{\,d}_{n(I,J)} \times \gamma^{d}_{n(I,J)} .$$

(*Prob_Eval.4*) The outputs from all clusters are summed, and the weighted mean is calculated as the trust value of node $n(K, L)$ known to node $n(I, J)$, $\lambda_{n(I,J)}^{n(K,L)}$:

$$\sum^{B} \gamma'''^{d}_{n(I,J)} \times \gamma^{d}_{n(I,J)} \Big/ \sum^{U} \gamma^{d}_{n(I,J)} \ ,$$

where $\{I,k\} \in \mathbb{Z}; \{J,L\} \in \mathbb{R}, J \neq L$.

The new ATR value of node $n(K, L)$ known to node $n(I, J)$ can be computed as

$$\hat{\gamma}_{n(I,J)}^{n(K,L)} = \alpha\left(\lambda_{n(I,J)}^{n(K,L)}\right) + \beta\left(\mu_{n(I,J)}^{n(K,L)}\right), \tag{1}$$

where α, β are the weight factors of the trust value and reputation value evaluated at each node, respectively, where $\alpha > \beta > 0$, and $\alpha + \beta = 1$.

To limit the querying time, node $n(I, J)$ must obtain the new ATR value of node $n(K, L)$ from these queries before the query timeout (t_Q) expires.

6.4.3.(3) ATR Update Process

After computing the ATR value, the system computes a ratio between the new ATR value ($\hat{\gamma}_{n(I,J)}^{n(K,L)}$) and the ATR value ($\gamma_{n(I,J)}^{n(K,L)}$) obtained from the ATR database. The ratio must be less than the difference threshold ε, defined in Equation 2, in order to accept the update, implying that $\hat{\gamma}_{n(I,J)}^{n(K,L)} \overset{change}{\Rightarrow} \gamma_{n(I,J)}^{n(K,L)}$; otherwise, the ATR update is dismissed.

$$\frac{\left|\hat{\gamma}_{n(I,J)}^{n(K,L)} - \gamma_{n(I,J)}^{n(K,L)}\right|}{\hat{\gamma}_{n(I,J)}^{n(K,L)}} \leq \varepsilon \tag{2}$$

The ratio is used to prevent an update of a fake reputation value received from the ATR query process. Then, the output ${\gamma''}^{b}_{n(I,J)}$ is recorded as the average ATR value of cluster b known to node $n(I, J)$,

$${\gamma''}^{b}_{n(I,J)} \overset{\text{change}}{\Rightarrow} \gamma^{b}_{n(I,J)}.$$

If the new ATR value of node $n(K, L)$ known to node $n(I, J)$ is below the *minimum ATR threshold* (ρ), the node $n(I, J)$ broadcasts to other cluster members of this untrustworthy $n(K, L)$. Those cluster members update the ATR value of node $n(K, L)$ known to them. Later, when node $n(K, L)$ initiates the probing process and wants any node in the cluster J to help forwarding its packets, such a request is rejected. Finally, the node with ATR value lower than the minimum ATR value is forced to retreat from the network because it will be ignored by other nodes. That is, no packet is being forwarded to this node and no other node forwards its packets.

- *Threat*: The malicious cluster member $n(I, J)$ can try to dump the ATR value of $n(K, L)$ known to other cluster members such that the new ATR value of $n(K, L)$ is severely decreased or below ρ. This problem is referred to as an insider attack, which will be discussed shortly.

6.4.4 Transmission Processes

In each transmission session, the source node first initiates the probing process to obtain crucial information from intermediate nodes to establish the path. Once such information is retrieved and the path is successfully evaluated, the source node then begins the data forwarding process to securely transmit the packets. After the destination node receives

the packets, the replying process starts, and the reply packets are traversed through ATAN to the source node anonymously. Various processes for data transmission in ATAN are illustrated in Figure 6.5.

All packets traversed in ATAN have virtually the same format, which consists of 9 fields: encrypted destination node's address, k_{Source}^{Seq}, trust requirement, sequence number, no-hop option, no-hash option, cluster ID, digests, and payload, as shown in Figure 6.6, except that the data packet and reply packet do not need k_{Source}^{Seq}, and the padding can be filled.

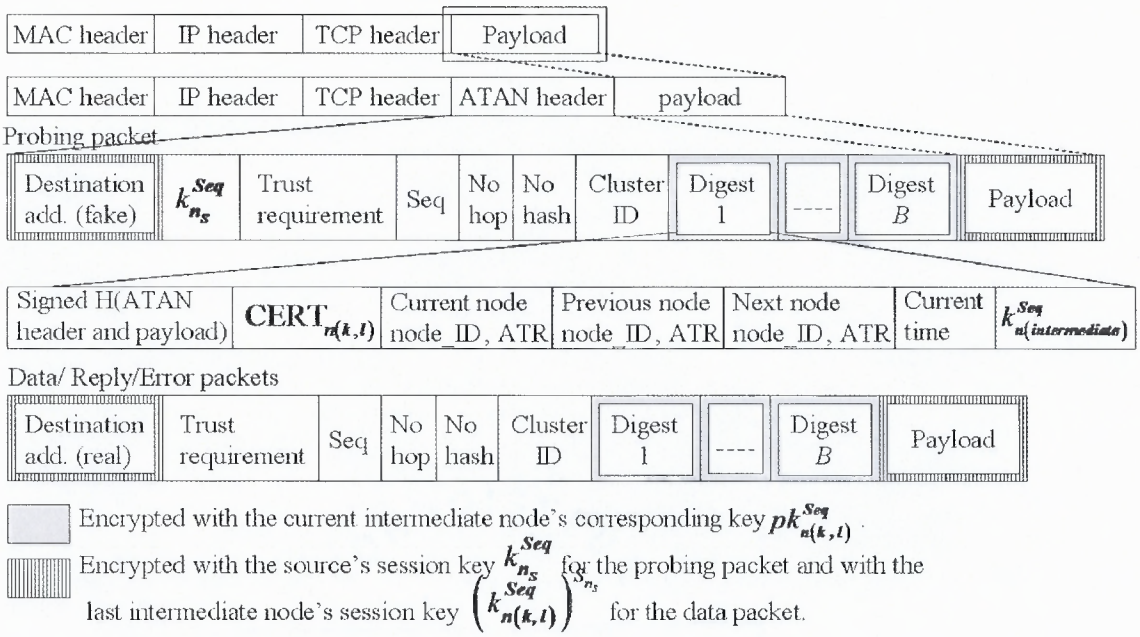


Figure 6.6 The packet format in ATAN [4].

6.4.4.(1) Probing Process

The source node n_S randomly selects a sequence number *Seq* and a session secret key, S_{n_S}, and computes the session public key as

$$k_{n_S}^{Seq} = \left(g^{S_{n_S}} \right) (\text{mod } p) .$$

At the beginning, the source sends out a probing packet without a given destination node. The Trust requirement (***T***) is defined as:

$$\boldsymbol{T} = \gamma_{n_S}^{n_S} \times \gamma_{n_S}^{C(n_S)} .$$

The Trust requirement depends on the ATR value of the source node and the average ATR value of the cluster to which the source node belongs. The ideas behind include the following: the Trust requirement determines the potential intermediate node during the ATR database lookup process. The source's packets will be forwarded by the intermediate node that has an ATR value equal to or lower than the Trust requirement. Thus, the trustworthiness of the path is improved and pre-determined. The probing process is described below and is illustrated in Figure 6.7.

(*Prob.1*) A current node (source or sender) $n(i,j)$ performs the ATR database lookup algorithm is to verify which next intermediate node $n(k,l)$ has the ATR value higher than Trust requirement (T).

a). If the Trust requirement condition fails and the *no-hop option* bit is zero (unset), the *packet rejection* algorithm with a reason of "no next hop", and notifies the source node with the error packet in Step (*Prob.4*).

b). If the Trust requirement condition fails but the *no-hop option* bit is one (set), the node $n(i,j)$ decreases the Trust requirement. This step is repeated until either the Trust requirement drops below half (proceed to Step 4) or the next intermediate node $n(k,l)$ is found (proceed to Step (*Prob.1.(c)*).

c). If the Trust requirement condition is met, the node performs the *packet accept* algorithm in Step (*Prob.7*).

(*Prob.2*) The intermediate node $n(k,l)$ performs the *repetition check* algorithm to check the repetition of the sequence number of both the previously received probing packets and the corresponding session public keys.

a) If both conditions are met, the intermediate node performs the packet rejection algorithm with the reason of "repeated packet", and notifies both the source node and the predecessor, who sent the two probing packets, using the error packet in Step (*Prob.4*).

b) If any condition fails, the intermediate node continues to Step (*Prob.3*).

(*Prob.3*) The intermediate node $n(k,l)$ computes two keys: $k^{Seq}_{n(k,l)}$ and $pk^{Seq}_{n(k,l)}$ corresponding to the sequence number, and stores the source node's session public key $k^{Seq}_{n_S}$ in the session key database for a period of time, τ. In addition, the intermediate node performs the *sender check* algorithm on the probing packet received to verify whether the sender node has the ATR value higher than the ATR threshold of the cluster l, $\overline{\gamma}_{C(l)}$, to which the intermediate node belongs.

a) If failed, the intermediate node notifies the source node with the reason of "bad sender" by the error packet in Step (*Prob.4*).

b) If passed, the intermediate node records some information about itself onto the available digest in the probing packet in Step (*Prob.5*).

(*Prob.4*) The intermediate node $n(k,l)$ performs the *packet rejection* algorithm, which rejects the probing packet with one of the following reasons, "no next hop", "repeated packet", or "bad sender". It provides some crucial information (called a digest) encrypted with the source node's session public key that came with the probing packet and replies to the source node.

(*Prob.5*) The intermediate node $n(k,l)$ performs the *ATR database lookup* algorithm, as in Anonymity bibliography [57], to verify which intermediate node has the ATR value higher than the Trust requirement. In this illustration, the selected node will become the next intermediate node $n(m,n)$

(*Prob.6*) The intermediate node $n(k,l)$ checks whether the *no-hash option* bit is set

a) If the *no-hash option* bit is one (set), the intermediate node skip to Step (*Prob.7*).

b) If the *no-hash option* bit is zero (unset), the intermediate node computes the hash function based on the SHA-1 scheme on the payload as $H(Payload)$.

(*Prob.7*) The intermediate node $n(k,l)$ performs the *packet accept* algorithm to write crucial information into the available digest.

In the *packet acceptance* algorithm, the intermediate node generates a random number S for $S \leq S_{MAX}$ and overwrites S consecutive digests with copies of encrypted crucial information.

(*Prob.8*) Steps (*Prob.2-Prob.7*) are repeated until all digests in the probing packet are filled. The last node initiates the replying phase and sends back the probing packet to the source

ALGORITHMS & COMMANDS

(1) ATR database lookup

$$\text{VERIFY}\left[(n(i,j),n(k,l))\middle|\forall(k,l),(k,l)\in \mathbb{U}, i,k\in \mathbb{N}, i=k, j,l\in \mathbb{M}, j\neq l : \gamma_{n(i,j)}^{n(k,l)} \geq T\right](\mathrm{M_P})$$

, where $T=\gamma_{n_S}^{n_S}\times\gamma_{n_S}^{C(n_S)}$

(2) Repetition check

$$\text{VERIFY}\left[(n(i,j),n(k,l))\middle|\forall(i,j), i\in \mathbb{N}, j\in \mathbb{M} : Seq(M_{P1})=Seq(M_{P2}),\ k_{sender}^{Seq}(M_{P1})=k_{sender}^{Seq}(M_{P2})\right](M_{P1},M_{P2})$$

$$Seq(M_{P2}),\ k_{sender}^{Seq}(M_{P1})=k_{sender}^{Seq}(M_{P2})\Big](M_{P1},M_{P2})$$

(3) Sender check

$$\text{VERIFY}\left[(n(i,j),n(k,l))\middle|(i,j)\in \mathbb{U}, i,k\in \mathbb{N}, i=k, j,l\in \mathbb{M}, j\neq l : \gamma_{n(k,l)}^{n(i,j)} \geq \overline{\gamma}_{C(l)}\right](\mathrm{M_P})$$

(4) $k_{n(k,l)}^{Seq}=\left(g^{R_{n(k,l)}}\right)(\mathrm{mod}\ p)$ and $pk_{n(k,l)}^{Seq}=\left(\left(k_{n_S}^{Seq}\right)^{R_{n(k,l)}}\right)(\mathrm{mod}\ p)=\left(g^{S_{n_S}R_{n(k,l)}}\right)(\mathrm{mod}\ p)$

Packet rejection

$$\text{REJECT}\left[(n(k,l),n(i,j))\middle|(i,j)\in \mathbb{U}, i,k\in \mathbb{N}, i=k, j,l\in \mathbb{M}, j=l : [\text{INFO}]\times\left[pK_{n_S}^{Seq}\right]^{\text{WRITE}}\Rightarrow[\text{BLOCK}]\right](\mathrm{M_E})$$

, where $[\text{INFO}]=\left[3, Seq, n(i,j), \gamma_{n(k,l)}^{n(i,j)}, n(k,l), \gamma_{n(k,l)}^{n(k,l)}, 0\right]$ for "no next hop" error,

$[\text{INFO}]=\left[2, Seq, n(i,j), \gamma_{n(k,l)}^{n(i,j)}, n(k,l), \gamma_{n(k,l)}^{n(k,l)}\right]$ for "repeated packet" error, and

$[\text{INFO}]=\left[1, Seq, n(i,j), \gamma_{n(k,l)}^{n(i,j)}, n(k,l), \gamma_{n(k,l)}^{n(k,l)}, \gamma_{n(k,l)}^{C(l)}, \overline{\gamma}_{C(l)}\right]$ for "bad sender" error.

(5) ATR database lookup

$$\text{VERIFY}\left[(n(k,l),n(m,n))\middle|\forall(m,n),(m,n)\in \mathbb{U}, k,m\in \mathbb{N}, k=m, l,n\in \mathbb{M}, l\neq n : \gamma_{n(k,l)}^{n(m,n)} \geq T\right](\mathrm{M_P})$$

, where $T=\gamma_{n_S}^{n_S}\times\gamma_{n_S}^{C(n_S)}$

(7) Packet acceptance

$$\text{ACCEPT}\left[(n(i,j),n(k,l))\middle|i,k\in \mathbb{N}, i=k, j,l\in \mathbb{M}, j=l : [\text{INFO}]\times\left[pK_{n_S}^{Seq}\right]^{\text{WRITE}}\Rightarrow[\text{BLOCK}]\right](\mathrm{M_P})$$

, where [INFO] = [H(T, Seq, no _ hop, no _ hash, reply _ block, and Payload),

$k_{n(k,l)}^{Seq}, n(i,j), \gamma_{n(k,l)}^{n(i,j)}, \overline{\gamma}_{C(j)}, n(k,l), \gamma_{n(k,l)}^{n(k,l)}, \overline{\gamma}_{C(l)}, n(m,n), \gamma_{n(k,l)}^{n(m,n)}, \overline{\gamma}_{C(n)}$, timestamp

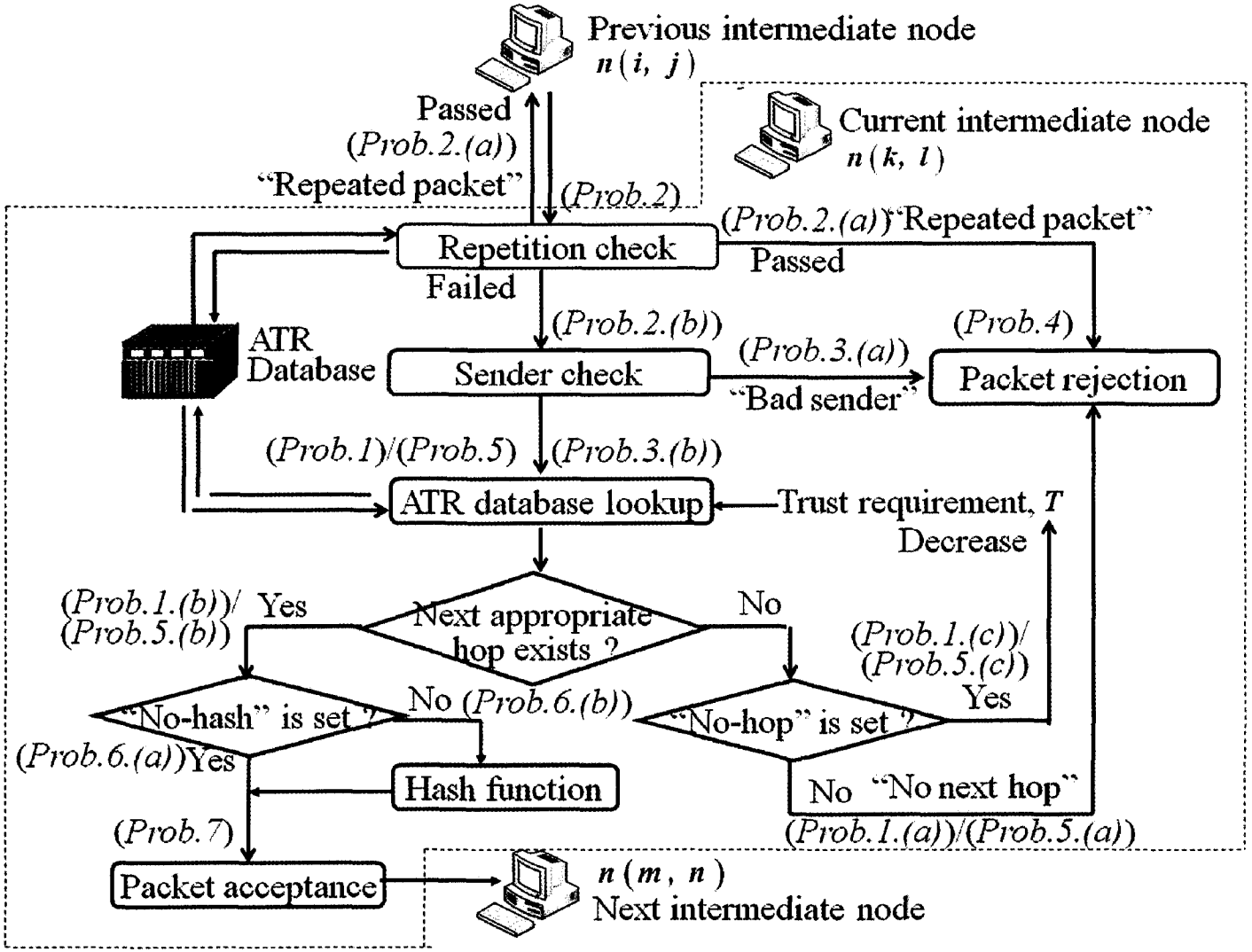


Figure 6.7 The forwarding steps in the probing process [4].

The *no-hop option* bit is inspected when the *ATR database lookup* process does not find an intermediate node which matches the Trust requirement. If the *no-hop option* bit is set off (by default), the packet is rejected and the error packet with the reason of "no next hop" is sent back to the source and the packet is dropped. If the *no-hop option* bit is set to one, the packet is forwarded to the node with the lower ATR value closest to the Trust requirement, but not less than, say, a half of the Trust requirement; otherwise, the error packet is sent and the packet is dropped as well.

To prevent the repetition of forwarding the packet twice, the intermediate node checks the sequence number of the received packet. Since the source node's identification is not placed in the probing packet, there might be a chance that multiple source nodes randomly select the same sequence number. Thus, the receiving intermediate node also checks the session public key. In case that two probing packets are generated from the same source node, they will have the same sequence number and the same session public key. Then, the receiving intermediate node accepts one probing packet, and rejects the other. The replayed packets can be dropped.

Each intermediate node provides information and places in the digest that includes a signed hash of the ATAN header and optional payload, a certificate of the current node, a current node ID, a previous node ID, a next node ID, their corresponding ATR values, an average ATR value of the cluster to which these three nodes belong, time stamp, and keys (only in reply to the probing packet).

- *Threat*: An adversary may alter the ATAN header such that next intermediate nodes unintentionally mistreat the packet. The digest is signed along with its certificate to protect the content.

The intermediate node *n(k, l)* also randomly selects a secret key $R_{n(k,l)}$ corresponding to the sequence number and source node's session key, and computes two associated keys based on the ELGamal cryptosystem:

$$ek_{n(k,l)}^{Seq} = \left(g^{R_{n(k,l)}}\right)(\text{mod } p) \text{ and}$$

$$pk_{n(k,l)}^{Seq} = \left(\left(k_{n_S}^{Seq}\right)^{R_{n(k,l)}}\right)(\text{mod } p)$$
$$= \left(\left(g^{S_{n_S}}\right)^{R_{n(k,l)}}\right)(\text{mod } p) = \left(g^{S_{n_S} R_{n(k,l)}}\right)(\text{mod } p) .$$

Only the keys $k_{n_S}^{Seq}$ and $R_{n(k,l)}$ will be kept in the node *n(k, l)*'s session key database for a period of τ. These keys will later be used by the intermediate node to encrypt the destination node's address in the data packet if this intermediate node is selected as the last receiver node. All this information in the digest will be encrypted with $pk_{n(k,l)}^{Seq}$ as $[\text{INFO}]\times\left[\boldsymbol{pk}_{n(k,l)}^{Seq}\right]$.

The intermediate node then records the following pair into the digest, denoted as

$$\left\{[\text{INFO}]\times\left[pk_{n(k,l)}^{Seq}\right], k_{n(k,l)}^{Seq}\right\}.$$

To ensure that the source can detect if the intermediate node has modified the payload, a payload is hashed by using a hash function. However, the hashed payload can incur a huge overhead; the source node may set the *no-hash option* bit, so that the intermediate node is not required to hash the payload. Anonymity is supposed to be higher if the no-hash option bit is unset and the no-hop option bit is set. However, the throughput of ATAN is expected to be higher if the no-hash option bit is set and the no-hop option bit is set.

Upon the receipt of the probing packet, the source node decrypts all digests and extracts information by computing the decrypting key

$$\left(\left(k_{n(k,l)}^{Seq}\right)^{S_{nS}}\right)(\text{mod } p)=\left(\left(g^{S_{n(k,l)}}\right)^{S_{nS}}\right)(\text{mod } p)$$
$$=\left(g^{S_{nS}S_{n(k,l)}}\right)(\text{mod } p)$$

and decrypting all encrypted digests as

$$\left([\text{INFO}]\times\left[pk_{n(k,l)}^{Seq}\right]\right)\Big/\left(k_{n(k,l)}^{Seq}\right)^{S_{nS}}$$
$$=\left([\text{INFO}]\times\left[g^{S_{n(k,l)}S_{nS}}\right]\right)\Big/\left(g^{S_{n(k,l)}}\right)^{S_{nS}}$$
$$=[\text{INFO}].$$

The source node evaluates information retrieved from the digest to justify whether the probing packet has been forwarded properly and honestly.

6.4.4.(2) Path Evaluation Process

To evaluate the path from the digests, the scores-based scheme is proposed. Let W be the set of all criteria defined by the source node, Y the set of criteria met by the intermediate node, and θ_v the score assigned to each criterion v, where $\sum_{v\in W}\theta_v = 1$.

The trustworthiness of the i^{th} intermediate node (ψ_i) is one if the sum of the met criteria exceeds the acceptance threshold (δ), and zero otherwise. That is,

$$\psi_i=\begin{cases}1, \textit{ if } \sum_{v\in Y}\theta_v \geq \delta\\ 0, \textit{ else}\end{cases}. \qquad (3)$$

Therefore, the source node can select which intermediate node is selected, as well as at which sequence, along the forwarding path.

A sample of the set of criteria and scores are illustrated in Table 5.3, but the importance and appropriateness of criteria and corresponding scores are left to be discussed in the future work.

6.4.4.(3) Data Forwarding Process

After the source node selects the intermediate nodes along the path, the last selected intermediate node becomes the last receiver node (n_L). Then, the source node encrypts each individual intermediate node ID (says node *n(m, n)*) with its predecessor's session key (says node *n(k, l)*) as

$$[n(m,\ n)\_\text{adress}]\times\left(\left(k_{n(k,\ l)}^{Seq}\right)^{S_{nS}}\right),$$

and writes into the digest corresponding to the node *n(k, l)*. This is similar to the onion routing method [66], [69], [71]; the only difference is that in ATAN each intermediate node's session key is obtained via a probing packet for a particular session. This can enhance secrecy of the session key.

In ATAN, all intermediate nodes' session keys are kept in the source's session key database in which each intermediate node keeps its secret session key (i.e., $R_{n(k,l)}$) and the source's session key ($k_{n_S}^{Seq}$) used for this source node per session.

The source node sends out the data packets in one session through the selected path to the destination node. If the source node does not initiate the data forwarding process or the data forwarding packet fails to reach the last receiver node before the time period τ expires, the last receiver node's symmetric session key is discarded, thereby terminating the data packet transmission.

Like the forwarding steps of the probing process, any intermediate node in the data forwarding process performs all steps 1-8 (in Figure 5.7), except Step (*Prob.5*), which is refined as:

(*Forward.5*) Node *n(k, l)* sorts the session key database, for the sequence number and $k_{n_S}^{Seq}$, and then computes $pk_{n(k,l)}^{Seq}$ to decrypt the associated digest to reveal which is the next intermediate node, denoted as

$$\left([n_D\_\text{address}] \times \left(k_{n(k,l)}^{Seq} \right)^{S_{nS}} \right) \Big/ pk_{n(k,l)}^{Seq}$$
$$= \left([n_D\_\text{address}] \times \left(g^{S_{n(k,l)}} \right)^{S_{nS}} \right) \Big/ g^{S_{n(k,l)} S_{nS}}$$
$$= [n_D\_\text{address}].$$

The selected node with n_D_address will become the next intermediate node *n(m, n)*.

Table 6.3 The Sample Set of Criteria and Scores [4]

Criterion (*v*)	Score
Report a "no-next-hop" error	0.0625
Report a "bad-predecessor" error	0
Report a "repeated-sequence-number" error	0
Being reported as a bad sender by the predecessor	0.0625
Being reported as a sender with "repeated-sequence-number" error by the predecessor	0.125
Any header filed is modified	0.125
A payload is modified	0.5

6.4.4.(4) Performance Evaluation Process

At the end of a session, all digests from all packets, including error packets if any, are collected and the performance is evaluated. The path evaluation process is utilized. If the trustworthiness of each intermediate node exceeds the acceptance threshold, the ATR value of that node known to the source is updated based on the additive increment and multiplicative decrement (AIMD) algorithm [77].

The new ATR value of node $n(K, L)$ known to the source $n(I, J)$ is calculated as:

$$\hat{\gamma}_{n(I,J)}^{n(K,L)} = \frac{\gamma_{n(I,J)}^{n(K,L)} + I}{D}, \tag{4}$$

where $I \geq 0$ is the additive increment factor, and $D \geq 1$ is the multiplicative decrement factor.

- If the node is trustworthy, $D = 1$ and $I > 0$.
- If the node is not trustworthy, any positive $D > 1$ and $I = 0$

Note that this new ATR value (from Equation 4) is not subject to the ratio test in Equation 2 in Subsection 6.4.3.(3). For example, $I = 1$ and $D = 2$.

This AIMD strategy ensures that the untrustworthy node's ATR value is rapidly decreased (multiplicative decrease) and the trustworthy node's ATR value is gradually increased (additive increase). The ATR evaluation is also applied to the error packet to adjust the ATR value of the maliciously acting node.

By helping forward the packets for the source, the intermediate node improves its ATR value known to the source. Later, all cluster members in the cluster to which the source belongs will do the local query to the source and then properly adjust the

intermediate node's ATR value known to them. Accordingly, the future packets from this intermediate node will be accepted and forwarded by these cluster members.

- *Threat*: The source may improperly update or ignore to update the ATR value of the trustworthy intermediate node. After the session ends, the intermediate node can execute the global ATR query to check if the ATR value has been updated, which can be indicated by the timestamp.

6.4.4.(5) Replying Process

The replying process is initiated when there is a packet to be sent back to the source node, which includes the probing, global, error and reply packets. There are two possible solutions in replying: one using the same forwarding path, and the other using the cluster ID. Many existing anonymous schemes use the reply block to contain the replying address of the source based on the onion routing [69]-[72]. However, this requires an extra space in the packet.

In ATAN, the replying path is simply routed through any intermediate nodes by using the cluster ID. The replying packet is forwarded among intermediate nodes until it reaches any cluster member associated with the cluster ID. That cluster member then broadcasts to other cluster members. Only the source can decrypt the payload. Although the cluster ID is very simple and decreases the space in the packet, the communication overhead increases from the multicast of the replying packet. Another disadvantage is that the adversary, which can either be a malicious destination or any insider or any outsider, can track the reply packet to the cluster to which the source belongs, though not exactly to the source node.

6.5 Threat Models

Here, the effectiveness of ATAN is analyzed and monitored when being deployed against the existing prevalent attacks.

1. Collusion attack

Two or more adversaries can collaborate with each other in several ways such as 1) by sharing information of all active nodes obtained such that all nodes in ATAN are under monitored, 2) by coverting the attacks from the user who attempts to detect any malicious activities, and 3) by organizing the attacking tasks such that the computation capability of the group of attackers is greatly exacerbated. ATAN may not sustain this attack if the group of attackers launches a correlation attack onto every member in a cluster to which the reply packet is sent (the cluster ID is openly known). One possible solution is to encrypt the cluster ID in layers but that may incur a huge packet size. The defense against such an attack will be more elaborated in the future.

2. Insider attack

The adversary can be either an insider from the same cluster or an insider from other clusters.

- If a cluster head has an administrative power to effectively control the cluster, a malicious cluster member may try to become a cluster head and operate the cluster with an ill intention. A voting protocol is proposed to reduce the administrative power held by the cluster head. A responsibility of some important incidents in the cluster is shared among members.
- A malicious cluster head may fake the ATR value of the target node during a global query process. A possible solution is to have the other cluster member of

the cluster to which the target node belongs queried to confirm whether the cluster head has given the ATR value correctly.

- A malicious cluster member (during local query and global query processes) may try to fake the ATR value of the target node. Three proposed solutions are: first, the ATR value of the target known to the source (querying node) is computed from the ATR value of the target known to the queried node, the ATR value of the queried node known to the source, and the average ATR value of the cluster to which the target belongs. Thus, the ATR value given by the malicious member is not directly used. Second, the weight factor for the reputation value is used to decrease the given ATR value. Third, the difference threshold is used to cancel an ATR update when there is a significant difference between the new ATR value and the stored one.

3. Traffic analysis attack

This type of attack can be divided into several types:

- Timing correlation attack. If the processing time of a particular intermediate node can be accurately estimated, the attacker can correlate two packets coming in and going out of that node. In ATAN, the forwarding steps in the probing and data forwarding processes require different computation timing, and therefore, ATAN effectively prevents this attack.
- Packet indistinguishability. If the data packet and the reply packet have different formats and have different sizes, the attacker can distinguish the type of packet and the direction of transmission. Since ATAN uses the virtually similar format

for each packet both in the forwarding and reversing paths, ATAN offers a minimal chance of being instigated by such an attack.

4. Logging attack

The malicious intermediate node keeps records of communication for every session that it helps forwarding. If the two end nodes continue their communication with more than one session, they will appear more often in the records than other nodes. Thus, the two nodes are likely linked as a source-destination pair, thus potentially exposing the anonymity. In ATAN, probing packets in different sessions are independently routed based on the varying Trust requirements. All intermediate nodes are also independently selected based on the varying ATR values. Thus, the same intermediate node may not likely be picked for the same source-destination pair, thus mitigating the threat of such an attack.

5. Denial of service (DoS) attack

ATAN executes the encryption and decryption based on the Diffie-Hellman cryptosystem at every intermediate node for every data packet. That may allow the malicious insider the best chance to launch the DoS attack by sending a load of normal or malicious packets onto trustworthy nodes. As a result, the trustworthy node's CPU and memory are fully exhausted. In this example, only colluded attackers can still operate so users are forced to send packets through a group of attackers. Thus, ATAN is likely susceptible to the DoS attack. By far, solutions using access control to limit the access and efficiently control the resource utilization may mitigate such attack.

6. Replay attack

An attacker sends a large number of packets that have been previously forwarded to the intermediate node in order to flood the node's memory to instigate the traffic analysis

attack (only one legitimate packet is in the memory and the rest are the replayed packets such that the packet is easily correlated). An attacker can also send a replayed packet to cause different results that may exploit the procedure's vulnerability. However, ATAN uses both sequence number and timestamp to effectively eliminate this replay attack.

6.6 Network Analysis

Since ATAN requires much information to be carried in the packet, one metric to evaluate the performance is the size of the management portion compared with the data payload portion. As illustrated in Figure 6.5, the probing packet carries destination address (32 bits), source's session key (~128-1024 bits, 512 bits in this example), trust (8 bits), sequence number (16 bits), no-hop option (1 bit), no-hash option (1 bit), B digests, and varying payload. Each digest consists of a hashed header (160 bits – SHA1 output), one cluster ID, three node_ID, three ATR value (each 8 bits), and time stamp (32 bits).

Consider N members in each cluster and M clusters in ATAN. Let X be the number of bits representing the cluster member ID, and Y representing the cluster ID. Thus, $X=log_2N$ and $Y=log_2M$. Therefore, each digest is (200+3X+Y)-bit long and the ATAN header in the probing packet has $570 + B(200+3X+Y)$ bits. Since the probing packet does not need to carry any payload, the size of the probing packet can be rather small. Similarly, there are $58 + B(200+3X+Y)$ bits in the ATAN header of the other three packet types. If there are 50 members in one cluster, 5 clusters in ATAN, and B is limited to 5, then the ATAN header of data packet is approximately 145 bytes. If the core network is Ethernet, the size of the maximum transmission unit (MTU) is 1500 bytes. Then, the ratio of the ATAN header and data payload is around 0.1, which is relatively small.

6.7 Conclusion

The loosely decentralized network management in ATAN does not require a central authority and knowledge of the whole network topology. This chapter demonstrates the deployability of ATAN in a decentralized network based on the trust and reputation model. The user joins the cluster based on its ATR value and the average ATR value of that cluster. The voting scheme is used to share the responsibility of managing the cluster among cluster members.

The trust and reputation system is designed to gradually increase the ATR value for a successful transmission but to sharply decrease the ATR value if a node acts maliciously. The probing packet obtains the candidate path and is used by the source to examine each intermediate node's trustworthiness. The source can choose the intermediate nodes of ATAN to help forward the packets. The intermediate node in turn inspects the source's trustworthiness to determine whether to help the source forward the packet.

This chapter has provided the framework for deploying ATAN, and future works include simulating ATAN by using a network simulator, network performance analysis, and a methodology for assigning the scores for each criterion (Table 6.3).

CHAPTER 7

CONCLUSION AND FUTURE WORK

This dissertation has presented fundamental attributes for evaluating mechanisms in providing one or more security services for GCSs as well as additional properties corresponding to those supported security services in Chapters 2 and 3. Moreover, this dissertation has presented known attacks that can severely disrupt or even shut down group communications in wired and wireless networks, and has presented necessary security requirements, and illustrated fundamental security services to meet these requirements and safeguard the communications against these attacks. It was demonstrated that several attacks can be prevented and mitigated by proposed security services. To complete the survey on SGC over wired and wireless networks, some open challenges that still need to be overcome are presented.

This dissertation has also incorporated ECC into GKM to decrease the key length while providing the same security level as that of other cryptosystems, and has adapted the cluster based key management scheme to make the ECC-GKM scheme more efficient. The group is separated into several clusters, which are independent from each other, particularly in the operations of key management. The cluster key is used to secure the group key selection and distribution while the group key is used to encrypt broadcast messages. When there is a membership change, the corresponding cluster key and the group key are changed in order to protect their secrecies. The periodic rekeying operation also strengthens the key secrecies.

The future work will provide extensive analysis on network performance (i.e., latency, bandwidth utilization, and throughput) and compare its performance against

other public key cryptosystems.

This dissertation has also demonstrated the deployability of ATAN in a decentralized network based on the trust and reputation model. The user joins the cluster based on its ATR value and the average ATR value of that cluster. The voting scheme is used to share the responsibility of managing the cluster among cluster members. The trust and reputation system is designed to gradually increase the ATR value for a successful transmission but to sharply decrease the ATR value if a node acts maliciously. The probing packet obtains the candidate path and is used by the source to examine each intermediate node's trustworthiness. The source can choose the intermediate nodes of ATAN to help forward the packets. The intermediate node in turn inspects the source's trustworthiness to determine whether to help the source forward the packet.

This chapter has provided the framework for deploying ATAN, and future works include simulating ATAN by using a network simulator, network performance analysis, and a methodology for assigning the scores for each criterion. Moreover, trust and reputation model will be modified such that network performance in terms of overheads can be improved. Security analysis will also be performed to demonstrate the effectiveness of incorporating trust and reputation into group communications systems.

APPENDIX

THE PROBABILITY MODEL

Following [78], the Beta probability distribution can be used to estimate some important features in this dissertation that may prevent or mitigate the impact from the attacks as follows:

1. The probability of the successful transaction at the next session in which node Y is part of the forwarding path.

Let n be the number of unsuccessful transactions originated from node X and forwarded by node Y in the past sessions,

p be the number of successful transactions originated from node X and forwarded by node Y in the past sessions,

θ be the true proportion of number of successful transactions for nodes X and Y,

$\hat{\theta}$ be the estimate for θ based on all past transactions originated from node X and forwarded by node Y.

Based on the Beta probability distribution, the prior probability distribution of $\hat{\theta}$, the probability distribution that represents a belief about an unknown quantity $\hat{\theta}$ before any observation results are recorded, is defined as

$$P(\hat{\theta}) = Beta(c_1, c_2) = \frac{\hat{\theta}^{C_1-1}(1-\hat{\theta})^{C_2-1}}{Beta(c_1, c_2)}.$$

where $0 \le \hat{\theta} \le 1$ and $c_1, c_2 > 0$. The Beta function can be defined by using the Gamma function as follows:

$$Beta(c_1, c_2) = \frac{\Gamma(c_1)\Gamma(c_2)}{\Gamma(c_1 + c_2)} = \int_0^1 \hat{\theta}^{c_1 - 1} \left(1 - \hat{\theta}\right)^{C_2 - 1} d\hat{\theta} .$$

The above equations are proven in Reference [78].

Assume that the probability of each transaction is independent of other transactions originated from node *X* and forwarded by node *Y*, and let *D* be the set of all transactions for the past sessions originated from node *X* and forwarded by node *Y*, and let $Tr_{XY}(S_i)$ be the variable representing a transaction for the i^{th} session originated from node *X* and forwarded by node *Y*. Thus,

$$D = \left\{ Tr_{XY}(S_1), Tr_{XY}(S_2), ..., Tr_{XY}(S_i) \right\}$$

If the transaction originated from node *X* and forwarded by node *Y* in the i^{th} session is successful, $Tr_{XY}(S_i) = 1$, otherwise $Tr_{XY}(S_i) = 0$ as follows:

$$Tr_{XY}(S_i) = \begin{cases} 1 & \textit{successful transaction;} \\ 0 & \textit{unsuccessful transaction.} \end{cases}$$

Let *L*(.) be the likelihood of having *p* successful transactions and *n* unsuccessful transactions; then, from the Beta probability distribution theory,

$$L\left(D \middle| \hat{\theta}\right) = \hat{\theta}^p \left(1 - \hat{\theta}\right)^n .$$

This likelihood is calculated, given the estimator $\hat{\theta}$, based on the assurance that the random variable *p* follows a binomial distribution as follows:

$$\Pr\left(p, n+p \middle| \hat{\theta}\right) = \binom{n+p}{p} \hat{\theta}^p \left(1 - \hat{\theta}\right)^n .$$

From the Bayes's theorem, the posterior probability distribution can be derived from the normalization of the prior probability distribution (or a normalizing constant) multiplied by the likelihood.

From the Bayes's theorem, with n and p fixed, and the prior distribution being Beta distribution $P(\hat{\theta}) = Beta(c_1, c_2)$, the posterior probability can be derived as

$$\Pr(\hat{\theta}|p,n) = \frac{\Pr(p,n|\hat{\theta})\Pr(\hat{\theta})}{\int_{\hat{\theta}} \Pr(p,n|\hat{\theta})\Pr(\hat{\theta})d\hat{\theta}}$$

$$= \frac{\hat{\theta}^p(1-\hat{\theta})^n \Pr(\hat{\theta})}{\int_{\hat{\theta}} \hat{\theta}^p(1-\hat{\theta})^n \Pr(\hat{\theta})d\hat{\theta}}.$$

That means the posterior is also a beta distribution $Beta(C1+c_1, C2+c_2)$.

Therefore, the posterior probability distribution is derived as follows:

$$\Pr(\hat{\theta}|D) = \frac{L(D|\hat{\theta})\Pr(\hat{\theta})}{\int_0^1 L(D|\hat{\theta})\Pr(\hat{\theta})d\hat{\theta}},$$

where $\hat{\theta} \in [0,1]$.

The integral in the denominator can be solved to get the posterior in a convenient form as proven in Reference [78].

Since it is already known that if $\Pr(\hat{\theta})$ is a Beta probability distribution with C_1 and C_2, one can assume that the posterior is also a Beta distribution with parameters $p+C_1, n+C_2$. That means,

$$\Pr(\hat{\theta}|D) = Beta(p+C_1, n+C_2).$$

From the Beta probability distribution, the expected value and variance of $Beta(C_1, C_2)$ are

$$E[Beta(C_1, C_2)] = \frac{C_1}{C_1 + C_2}, \text{ and } VAR[Beta(C_1, C_2)] = \frac{C_1 C_2}{(C_1 + C_2)^2 (C_1 + C_2 + 1)},$$

respectively.

Thus, the expected value and variance of $Beta(p + C_1, n + C_2)$ are

$$E[Beta(p + C_1, n + C_2)] = \frac{p + C_1}{C_1 + C_2 + n + p},$$

$$\text{and } VAR[Beta(p + C_1, n + C_2)] = \frac{(p + C_1)(n + C_2)}{(C_1 + C_2 + n + p)^2 (C_1 + C_2 + n + p + 1)}.$$

Note that from the Law of the Total Probability, the prior probability of $\hat{\theta}$ is equal to the prior expected value of the posterior probability of $\hat{\theta}$. Thus, for any random variable, p,

$$\Pr(\hat{\theta}) = E[\Pr(\hat{\theta} | p)],$$

where $\Pr(\hat{\theta} | p)$ is the conditional probability of $\hat{\theta}$ given p.

One can estimate the parameters C_1 and C_2 as follows:

$$C_1 = E(\hat{\theta}) \left(\frac{E(\hat{\theta})(1 - E(\hat{\theta}))}{VAR(\hat{\theta})} - 1 \right) \text{ and } C_2 = (1 - E(\hat{\theta})) \left(\frac{E(\hat{\theta})(1 - E(\hat{\theta}))}{VAR(\hat{\theta})} - 1 \right).$$

Therefore, the parameters C_1 and C_2 can be derived in the same way. These parameters change as the transaction grows.

Reference [78] has proven that to determine the estimate of the probability that the next transaction is successful in which node X originates and node Y forwards, based on given past records of $n+p$ sessions, one can derive $\Pr\left(Tr_{XY}^{Si+1}=1|D\right)$ as follows:

$$\mathrm{P}\left(Tr_{XY}^{Si+1}=1|D\right)=\int_{\hat{\theta}_{n+p}}\mathrm{P}\left(Tr_{XY}^{Si+1}=1|\hat{\theta}_{n+p},D\right)\mathrm{P}\left(\hat{\theta}_{n+p},D\right)d\hat{\theta}_{n+p},$$

where $\hat{\theta}_{n+p}$ is the estimated successful proportion based on $n+p$ previous transactions. $\Pr\left(Tr_{XY}^{Si+1}=1|\hat{\theta}_{n+p},D\right)$ is the likelihood for $Tr_{XY}^{Si+1}=1$ given the estimated $\hat{\theta}_{n+p}$ from $n+p$ previous transactions.

Therefore, the probability of the successful transaction at the next session in which node Y is part of the forwarding path can be calculated from

$$\Pr\left(Tr_{XY}^{Si+1}=1|D\right)=E\left(\hat{\theta}_{n+p}|D\right)$$

$$E\left[\hat{\theta}_{n+p}|D\right]=\frac{p+C_1}{C_1+C_2+n+p}.$$

2. The probability of the true recommendation at the next request in which node Y would reply the trustworthiness request sent by node X regarding a node of interest Z.

Let n be the number of false recommendations sent by node X and replied by node Y regarding node Z's reputation in the past requests,

p be the number of true recommendations sent by node X and replied by node Y regarding node Z's reputation in the past requests,

θ be the true proportion of number of true recommendations between nodes X and Y regarding the node of interest Z,

$\widehat{\theta}$ be the estimate for θ based on all past recommendations sent by node X and replied by node Y regarding node Z's reputation.

Assume that the probability of each recommendation is independent of other recommendations sent by node X and replied by node Y regarding the node of interest Z, and let G be the set of all recommendations for the past sessions between nodes X and Y, and let $Tr_{XY|Z}(S_i)$ be the variable representing a recommendation for the i^{th} request sent by node X and replied by node Y regarding the node of interest Z. Thus,

$$G = \left\{ Tr_{XY|Z}(S_1), Tr_{XY|Z}(S_2), ..., Tr_{XY|Z}(S_i) \right\}$$

If the recommendation replied at the i^{th} request is true, $Tr_{XY|Z}(S_i) = 1$, otherwise $Tr_{XY|Z}(S_i) = 0$ as follows:

$$Tr_{XY|Z}(S_i) = \begin{cases} 1 & \textit{true recommendation;} \\ 0 & \textit{false recommendation.} \end{cases}$$

Therefore, using the same probability model as discussed earlier, the probability of the true recommendation replied by node Y at the next request can be calculated from

$$\Pr\left(Tr_{XY|Z}^{Si+1} = 1 | G \right) = E\left(\widehat{\theta}_{n+p} | G \right)$$

$$E\left[\widehat{\theta}_{n+p} | G \right] = \frac{p + C_1}{C_1 + C_2 + n + p}.$$

REFERENCES

1. Sakarindr, P., and Ansari, N., "Security services on group communications," *IET Information Security Special Issue on Multi-Agent and Distributed Information Security*, June 2010.

2. Sakarindr, P., and Ansari, N., "Security services in group communications over wireless infrastructure, mobile ad-hoc, and wireless sensor network," *IEEE Wireless Communication Magazine, Special Issue on Security in Wireless Mobile Ad Hoc and Sensor Networks*, vol. 14, no.5, pp. 8-20, Oct. 2007.

3. Sakarindr, P., and Ansari, N., "Elliptic curve cryptosystem-based group key management on secure group communications," *IEEE Military Communications Conference MILCOM 2007*, Orlando, FL, pp. 1-6, Oct. 2007.

4. Sakarindr, P., and Ansari, N., "Adaptive trust-based anonymous network," *International Journal of Security and Networks, Special Issue on Computer and Network Security*, vol. 2, no. 1/2, pp.11-26, Mar. 2007.

5. O. Pereira and Q. Jean-Jacques, "Some attacks upon authenticated group key agreement protocols," *Journal of Computer Security*, vol. 11, no. 4, pp. 555-580, Jul. 2003.

6. M. Waldvogel, G. Caroni, D. Sun, N. Weiler, and B. Plattner, "The VersaKey framework: versatile group key management," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 9, pp. 1614-1631, Sep. 1999.

7. S. Banerjee and B. Bhattacharjee, "Scalable secure group communication over IP multicast," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 8, pp. 1511-1527, Oct. 2000.

8. C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16-30, Feb. 2000.

9. Y. Amir, Y. Kim, C. N. Rotaru, J. L. Schultz, J. Stanton, and G. Tsudik, "Secure group communication using robust contributory key agreement", *IEEE Transactions on Parallel and Distributed Systems TPDS 2004*, vol. 15, no. 5, pp. 468-480, May 2004.

10. Y. Sun and K. J. R. Liu "Hierarchical group access control for Secure Multicast Communications," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1514-1526, Dec. 2007.

11. Q. Zhang and Y. Wang, "A centralized key management scheme for hierarchical access control," *IEEE Global Telecommunications Conference GLOBECOM 2004*, Dallas, Texas, pp. 2067-2071, Dec. 2004.

12. Z. Chen, J. Huang, D. Huang, Z. Jianhong, and W. Yumin, "Provably secure and ID-based group signature scheme," *Proceedings of the Eighteenth International Conference on Advanced Information Networking and Applications AINA 2004*, Fukuoka, Japan, vol. 2, pp. 384-387, Mar. 2004.

13. N.-Y. Lee, "Threshold signature scheme with multiple signing policies," *IEE Proceedings Computers and Digital Technologies*, vol. 148, no. 2, pp. 95-99, Mar. 2001.

14. G. Ateniese, J. Camenisch, M. Joyce, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," *Proceedings of the Twentieth Annual International Cryptology Conference CRYPTO 2000*, Santa Barbara, CA, pp. 255-270, Aug. 2000.

15. L. Xiao, Y. Liu, W. Gu, D. Xuan, and X. Liu, "A design of overlay anonymous multicast protocol," *Proceedings of the Twentieth International Parallel and Distributed Processing Symposium IPDPS 2006*, Rhodes Island, Greece, Apr. 2006.

16. P. F. Syverson, D. M. Goldschlag, and M. G. Reed, "Anonymous connections and onion routing," *IEEE Journal on Selected Areas in Communications*, vol. 16, no.4, pp. 44-53, May 1998.

17. M. K. Reiter and A. D. Rubin, "Crowds: anonymity for web transactions," *ACM Transactions on Information and System Security TISSEC 1998*, vol. 1, no. 1, pp. 66-92, Nov. 1998.

18. C. Grosch, "Framework for anonymity in IP-multicast environments," *IEEE Global Telecommunications Conference GLOBECOM 2000*, San Francisco, CA, vol. 1, pp. 365-369, Dec. 2000.

19. S. Dolev and R. Ostrovsky, "Xor-trees for efficient anonymous multicast and reception," *ACM Transactions on Information and System Security TISSEC 2000*, vol. 3, no. 2, pp. 63-84, May 2000.

20. C. Shields and J. J. Garcia-Luna Aceves, "KHIP: a scalable protocol for secure multicast routing," *ACM SIGCOMM 1999 Computer Communication Review*, Cambridge, MA, vol. 29, no. 4, pp. 53-64, Sep. 1999.

21. Y.-C. Shim, "A new approach for secure multicast routing in a large scale network", *The Third International Conference on Information and Communications Security ICICS 2001*, Xian, China, pp. 95-106, Nov. 2001.

22. T. Li and K.-Y. Lam, "A secure group solution for multi-agent EC system," *Proceedings of the Fifteenth International Parallel and Distributed Processing Symposium IPDPS 2001*, San Fransaico, CA, pp. 1749-1756, Apr. 2001.

23. S. Shin, H. Fathi, K. Kobara, and H. Imai, "A secure group communication framework in private personal area networks (P-PANs)," *The Third International Conference on Wireless and Mobile Communications ICWMC 2007*, Guadeloupe, French Caribbean, pp. 59-67, Mar. 2007.

24. T. Hardjono and B. Weis, "The multicast group security architecture", IETF working group, multicast security (msec) chapter, retrieved from http://www.ietf.org/html.charters/msec-charter.html and http://www.ietf.org/rfc/rfc3740.txt, accessed December 2009.

25. G. Chaddoud and V. Varadharajan, "Efficient secure group management for SSM," *IEEE International Conference on Communications ICC 2004*, Paris, France, pp. 1436-1440, Jun. 2004.

26. Gupta, S. K. S., and Cherukuri, S., "An adaptive protocol for efficient and secure multicasting in IEEE 802.11 based wireless LANs," *IEEE Wireless Communications and Networking WCNC 2003*, New Orleans, LA. vol. 3, pp. 2021-2026, Mar. 2003.

27. Wadaa, A., Olariu, S., Wilson, L., Eltoweissy, M., and Jones, K., "On providing anonymity in wireless sensor networks," *Proceedings of the Tenth International Conference on Parallel and Distributed Systems ICPADS 2004*, Newport Beach, CA, pp. 411-418, Jul. 2004.

28. Karlof, C., and Wagner, D., "Secure routing in wireless sensor networks: attacks and countermeasures," *Elsevier AdHoc Networks Journal, Special Issue on Sensor Network Applications Protocols*, vol. 1, no 2-3, pp. 293-315, Sep. 2003.

29. Maheshwari, R., Gao, J., and Das, S. R., "Detecting wormhole attacks in wireless networks using connectivity information," *The Twenty-Sixth Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2007*, Anchorage, AK, pp. 107-115, May 2007.

30. Decleene, B., Dondeti, L., Griffin, S., Hardjono, T., Kiwior, D., Kurose, J., Towsley, D., Fasudevan, S., and Zhang, C., "Secure group communications for wireless networks," *IEEE Military Communications Conference MILCOM 2001*, Washington D.C., vol. 1, pp. 113-117, Oct. 2001.

31. Zhu, S., Setia, S., and Jajodia, S., "LEAP: efficient security mechanisms for large-scale distributed sensor networks," *Proceedings of the Tenth ACM Conference on Computer and Communications Security CCS 2003*, Washington D.C., pp. 62-72, Oct. 2003.

32. Yu, Z., and Guan, Y., "A robust group-based key management scheme for wireless sensor networks," *IEEE Wireless Communications and Networking WCNC 2005*, New Orleans, LA, vol. 4, pp. 1915-1920, Mar. 2005.

33. Zhang, W., and Cao, G., "Group rekeying for filtering false data in sensor networks: a predistribution and local collaboration-based approach," *The Twenty-fourth Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2005*, Miami, FL, vol. 1, pp. 503-514, Mar. 2005.

34. Striki, M., and Baras, J., "Towards integrating key distribution with entity authentication for efficient, scalable and secure group communication in MANETs," *The IEEE International Conference on Communications ICC 2004*, Paris, France, vol. 7, pp. 4377-4381, Jun. 2004.

35. Balachandran, R. K., Ramamurthy, B., Zou, X., and Vinodchandran, N. V., "CRTDH: an efficient key agreement scheme for secure group communications in wireless ad hoc networks," *The IEEE International Conference on Communications ICC 2005*, Seoul, Korea, vol. 2, pp. 1123-1127, May 2005.

36. Sun, Y., Trappe, W., and Ray Liu, K. J., "A scalable multicast key management scheme for heterogeneous wireless networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 653-666, Aug. 2004.

37. Westerhoff, L., Reinhardt, S., Schafer, G., and Wolisz, A., "Security analysis and concept for the multicast-based handover support architecture MOMBASA," *IEEE Global Telecommunications Conference GLOBECOM 2004*, Dallas, TX, vol. 4, pp. 2201- 2207, Nov. 2004.

38. Kaya, T., Lin, G., Noubir, G., and Yilmaz, A., "Secure multicast groups on Ad Hoc networks," *Proceedings of the First ACM Workshop on Security of Ad Hoc and Sensor Networks SASN 2003*, Fairfax, VA, pp. 94-103, Oct. 2003.

39. Lazos, L., and Poovendran, R., "Cross-layer design for energy-efficient secure multicast communications in ad hoc networks," *The IEEE International Conference on Communications ICC 2004*, Paris, France, vol. 6, pp. 3633- 3639, Jun. 2004.

40. Huang, J.-H., Buckingham, J., and Han, R., "A level key infrastructure for secure and efficient group communication in wireless sensor networks," *The First International Conference on Security and Privacy for Emerging Areas in Communications Networks SECURECOMM 2005*, Athens Greece, pp. 249-260, Sept. 2005.

41. Whee, K. D., Seung, L. J., Jong, K. W., and Jung, E., "An efficient LKH tree balancing algorithm for group key management," *IEEE Communications Letters*, vol. 10, no. 3, pp. 222–224, Mar. 2006.

42. Kim, Y., Perrig, A., and Tsudik, G., "Group key agreement efficient in communication," *IEEE Transactions on Computers*, vol. 53, no. 7, pp. 905–921, Jul. 2004.

43. Zhang, Q., and Wang, Y., "A centralized key management scheme for hierarchical access control," *IEEE Global Telecommunications Conference GLOBECOM 2004*, Dallas, TX, vol.4, pp. 2067–2071, Nov. 2004.

44. Sencun, Z., Setia, S., and Jajodia, S., "Performance optimizations for group key management schemes," *Proceedings of the Twenty-Third International Conference on Distributed Computing Systems*, Providence, RI, pp. 163–171, May 2003.

45. Waldvogel, M., Caronni, G., Dan, S., Weiler, N., and Plattner, B., "The VersaKey framework: versatile group key management, *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 9, pp. 1614–1631, Sep. 1999.

46. Yasinsac, A., Thakur, V., Carter, S., and Cubukcu, I., "A Family of Protocols for Group Key Generation in Ad Hoc Networks," *The IASTED International Conference on Communications and Computer Networks IASTED CCN 2002*, Cambridge, MA, pp. 183-187, Nov. 2002.

47. Zou, X., Ramamurthy, B., and Magliveras, S. S., *Secure Group Communications over Data Networks*, Science and Business Media, 1st ed. Santa Clara, CA: Springer-Verlag TELOS, 2005.

48. Washington, L. C. (2003). *Elliptic Curves: Number Theory and Cryptography*, 1st ed. Florida: Chapman & Hall/CRC A CRC Press, 2003.

49. Miller, V. S., "Uses of Elliptic Curve in Cryptography," *Proceedings of the Fifth Annual International Cryptology Conference on Advances in Cryptology CRYPTO 1985*, Lecture Notes in Computer Sciences, vol. 218, New York: Springer-Verlag, 1986, pp. 417-426.

50. Koblits, N., *Introduction to Elliptic Curves and Modular Forms*, 2nd ed. New York: Springer-Verlag, 1993.

51. Sutikno S., and Surya, A., "An architecture of F(22N) multiplier for elliptic curves cryptosystem," *Proceedings of the IEEE International Symposium on Circuits and Systems ISCAS 2000*, Geneva, Switzerland, vol. 1, pp. 279–282, May 2000.

52. Zhi, L., Higgins, J., and Clement, M., "Performance of finite field arithmetic in an elliptic curve cryptosystem," *Proceedings of the Ninth Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems MASCOTS 2001*, Cincinnati, OH, pp. 249–256, Aug. 2001.

53. Botes, J. J., and Penzhorn, W. T., "An implementation of an elliptic curve cryptosystem," *Proceedings of the 1994 IEEE South African Symposium on Communications and Signal Processing COMSIG 1994*, Stellenbosch, South Africa, pp. 85–90, Oct. 1994.

54. De Mulder, E., Buysschaert, P., Ors, S. B., Delmotte, P., Preneel, B., Vandenbosch, G., and Verbauwhede, I., "Electromagnetic analysis attack on an FPGA implementation of an elliptic curve cryptosystem," *The International Conference on Computer as a Tool IEEE EUROCON 2005*, Glasgow, Scotland, vol. 2, pp. 1879– 1882, Nov. 2005.

55. Samoa, K.S., Semay, O., and Takagi, T., "Analysis of fractional window recoding methods and their application to elliptic curve cryptosystems," *IEEE Transactions on. Computers*, vol. 55, no. 1, pp. 48-57, Jan. 2006.

56. Raju G. V. S., and Akbani, R., "Elliptic curve cryptosystem and its applications," *The Proceedings of the 2003 IEEE International Conference on Systems, Man, and Cybernetics SMC* 2003, Washington D.C., vol. 2, pp. 1540 – 1543, Oct. 2003.

57. Anonymity bibliography, "Researches on anonymity", Retrieved on July 18, 2005 from the World Wide Web http://www.freehaven.net.

58. Helmers, S., "A brief history of anon.penet.fi: The legendary anonymous remailer," Retrieved October 10, 2005 from the World Wide Web: http://www.december.com/cmc/mag/1997/sep/helmers.html.

59. Guan, Y., Fu, X., Bettati, R., and Zhao, W., "An optimal strategy for anonymous communication protocols," *Proceedings of the Twenty-Second International Conference on Distributed Computing Systems ICDCS 2002*, Vienna, Austria, pp. 257 – 266, Jul. 2002.

60. Rennhard, M., Rafaeli, S., Mathy, L., Plattner B., and Hutchison, D., "An architecture for an anonymity network," *Proceedings of the Tenth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises WET ICE 2001*, Cambridge, MA, pp. 165-170, Jun. 2001.

61. Chaum, D., "Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, vol. 24, no. 2, pp. 84-88, Feb. 1981.

62. Reiter, M., and Rubin, A., "Crowd: anonymity for web transactions," *ACM Transactions on Information and System Security TISSEC 1998*, vol. 1, no. 1, pp. 66-92, Nov. 1998.

63. Liang, Z., and Shi, W., "PET: A personalized trust model with reputation and risk evaluation for P2P resource sharing," *Proceedings of the Thirty-Eighth Annual Hawaii International Conference on System Sciences HICSS 2005*, Hawaii, HI, vol. 7, pp. 1-10, Jan. 2005.

64. Singh, A., and Liu, L., "TrustMe: anonymous management of trust relationships in decentralized P2P systems," *Proceedings of the Third International Conference on Peer-to-Peer Computing P2P 2003*, Linköping, Sweden, pp. 142-149, Sep. 2003.

65. Dingledine, R., Mathewson, N., and Syverson, P., "Reputation in P2P Anonymity Systems," *First Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, Jun. 2003.

66. Wang, Y., and Vassileva, J., "Trust and reputation system in peer-to-peer networks," *Proceedings of the Fifth International Conference on Peer-to-Peer Computing P2P 2005*, Konstanz, Germany, pp. 150-157, Aug. 2005.

67. Marti, S., and Garcia-Molina, H., "Identity crisis: anonymity VS. reputation in P2P systems," *Proceedings of the Third International Conference on Peer-to-Peer Computing P2P 2003*, Linköping, Sweden, pp. 134-141, Sep. 2003.

68. Freedman, M. J., and Morris, R., "Tarzan: A peer-to-peer anonymizing network layer," *Proceedings of the Ninth ACM Conference on Computer and Communications Security CCS* 2003, Washington D.C., pp. 193-206, Oct. 2003.

69. Rennhard, M., and Plattner, B., "Introducing MorphMix: Peer-to-peer based anonymous Internet usage with collusion detection," *Proceedings of the 2002 ACM Workshop on Privacy in the Electronic Society WPES 2002*, Washington D.C., pp. 91-102, Nov. 2002.

70. Camenisch, J., and Lysyanskaya, A., "A Formal Treatment of Onion Routing," *Proceedings of the Twenty-Fifth Annual International Cryptology Conference on Advances in Cryptology CRYPTO 2005*, Santa Barbara, CA, pp. 169-187, Aug. 2005.

71. Dingledine, R., Mathewson, N., and Syverson, P., "Tor: The second generation onion router," *Proceedings of the Thirteenth USENIX Security Symposium SECURITY 2004*, San Diego, CA, pp. 303-320, Aug. 2004.

72. Goldschlag, D., Reed, M., and Syverson, P., "Onion routing for anonymous and private internet connections," *Communications of the ACM*, vol. 42, no. 2, pp. 39-41, Feb. 1999.

73. Danezis, G., Dingledine, R., and Mathewson, N., "Mixminion: Design of a type III anonymous remailer protocol," *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, pp. 2-15, May 2003.

74. Diffie, W., and Hellman, M. E., "New direction in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no.6, pp. 644-654, 1976.

75. ElGamal, T., "A public key cryptosystem and a signature scheme based on discrete logarithms" *IEEE Transactions on Information Theory*, vol. 31, pp. 469-472, 1985.

76. Steiner, M., Tsudik, G., and Waider, M., "Diffie-Hellman key distribution extended to group communications," *Proceedings of the Third ACM Conference on Computer and Communications Security CCS 1996*, New Delhi, India, pp. 31-37, Mar. 1996.

77. Lahanas, A., and Tsaoussidis, V., "Additive Increase Multiplicative Decrease - Fast Convergence (AIMD-FC)," *Proceedings of the Joint International Conference on Wireless Networks*, pp. 511-522, Aug. 2002.

78. Mui, L., "Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks," Ph.D. Dissertation, Massachusetts Institute of Technology, 2003.