ERRATA

p 15 line 22 equation 2.1: replace " $Pr(Al'(f(x),1^k) \in f^{-1}(f(x)) < \frac{1}{p(k)}$ " with " $Pr(Al'(f(x),1^k) \in f^{-1}(f(x))) < \frac{1}{p(k)}$ ", thus adding a new closing parenthesis.

p 21 line 23: replace " $\{K_{18}^{'}\}K_{48}$ " with " $\{K_{18}^{'}\}K_{58}$ "

p 53 line 15: replace "u_1, ..., u_n;" with "U_1, ..., U_n;"

p 53 line 16: replace "s_1, ..., s_m;" with "S_1, ..., S_m;"

p 57 lines 8 and 9: replace "U_{i1}, …, U_{ik}" with " $U_{i_1},...,U_{i_k}$ "

p 57 line 9 (equation 3.1) : replace " $0 \le k \le n$ " with " $0 < k \le n$ "

p 57 lines 23 and 24: replace "S_{j1}… S_{jl}" with " $S_{j_1},...,S_{j_l}$ "

p 57 line 24 (equation 3.2) : replace " $0 \le l \le m$ " with " $0 < l \le m$ "

p 65 line 8 equation 3.8: replace " $GA \subseteq GA_1 + GA_2, GA_1 \in TUR, GA_2 \in TSR$ " with " $GA \subseteq TUR \times TSR$ "

p 65 line 14 equation 3.9: replace " $GA^* \subseteq GA_1 + GA_2, GA_1, GA_2 \in TSR \cup TUR$ " with " $GA^* \subseteq TR \times TR, TR = TUR \cup TSR$ "

p 66 line 11 equation 3.10: replace " $GA^* = (S_1 \to SG_1) \vee (S_2 \to SG_2)$ " with " $GA^* = (S_1 \to SG_1) \times (S_2 \to SG_2)$ "

p 88 line 19 (row 15 table 4.1): replace "A polynomial function" with "A positive polynomial statement"

p 89 lines 12-16: replace the second, third and fourth sentences with "For every positive polynomial statement *p*(.) and for all sufficiently large *k*, the probability of correctly guessing the current dynamic key DK_c from the previous dynamic keys DK_1 … DK_{c-1} is as follows:"

p 89 line 17 equation 4.2: replace " $\forall i, c \in N, 1 \le i \le c, Pr(Al(\{DK_i\}) = DK_c) \le \frac{1}{2^s}$ " with " $\forall c \in N, Pr(Al(DK_1,...,DK_{c-1}) = DK_c) \le \frac{1}{p(k)}$ "

p 90 line 10 equation 4.4: replace " $SK = IK + TK_1 + ... + TK_{m-1} + TK_m$ " with " $SK = (IK + TK_1 + ... + TK_{m-1} + TK_m) \mod 2^s$ "

p 91 line 2: delete "taking *m*+1 parameters"

p 91 line 5: replace "parameters" with "input keys"

p 92 line 7 equation 4.9: replace " $Pr(Al(\{DK_i\}) = DK_c) = Pr(Al(.) = SK))$ " with " $Pr(Al(DK_1,...,DK_{c-1}) = DK_c) = Pr(Al'(DK_1,...,DK_{c-1}) = SK)$ "

p 93 line 3 equation 4.13: replace " $Pr(Al(.) = SK) \le \frac{1}{2^s}$ " with " $Pr(Al'(DK_{c-m-1},...,DK_{c-1}) = SK) \le \frac{1}{p(k)}$ "

p 93 line 5 equation 4.14: replace " $Pr(Al(\{DK_i\}) = DK_c) < \frac{1}{2^s}$ " with " $Pr(Al(DK_1,...,DK_{c-1}) = DK_c) = Pr(Al'(DK_1,...,DK_{c-1}) = SK) \le \frac{1}{p(k)}$ "

p 93 line 19: replace "unique" with "identical for all involved parties"

p 94 line 1 replace the proof of lemma 4.3 with:

> "By assumption of lemma 4.1, function *f*(.) is a one-way function. Thus, each input of *f*(.) is associated with only one output value. Because involved parties share the same function *f*(.) to generate dynamic key

sequences and for the same input TK_1, TK_2, …, TK_m and IK, the produced dynamic keys are identical. By sharing the same sequence of dynamic keys, involved parties are able to encrypt and decrypt communication messages."

p 94 line 11: replace "By replacing a one-way hash $h(.)$ with the function $f(.)$" with "By replacing function $f(.)$ with a one-way hash function $h(.)$"

p 95 figure 4.2 top labels: replace "*EK"* and "*IK"* with " EK' " and " IK' "

p 98 lines 1 and 2 replace "$\mathbb{R}$" with "$\mathbb{R}_{EK}$"

p 107 lines 16 and 17: replace "no leader-candidate is available;" with "a replacement leader-candidate is available;"

p 122 line 21: replace "as an authentication request to *u*" with "as an authentication request to *s*"

ADDENDUM

p 15 line 21: insert before ":" at the end of the sentence ", the probability $Pr(.)$ to find the reverse function of $f(x)$ is as follows".

p 15 line 22 equation (2.1): Comment: The equation explains that for any positive polynomial statement $p(k)$, it is infeasible to find a polynomial algorithm *Al'* to compute the reverse function $f^{-1}(f(x))$

p 51 line 25: Comment: The other works on wireless networks [BCEP04, AST00, Han00] named in the section are not reviewed in chapter 2 because they are extensions of the Kerberos authentication model. The WEP and WPA2 authentication protocols are also not mentioned in chapter 2 because they are in a lower level (network level). The scope of the thesis applies to wireless network users and services in middleware and application levels (for example, P2P and Cloud Computing).

p 57: Insert at the end of the page:

"Let *y* be the number of service groups in the system. The set of all service groups in the system is written as $P=\{SG_1, \ldots, SG_y\}$."

The definition of the set of all service groups, P, is used to define group authentication relationship in pages 64 and 65.

p 88 table 4.1 row 11, After "polynomial function" insert " (supposed to be a one-way hash function) "

p 88 table 4.1: At the end of table insert :

$\{X\}K$	X is encrypted using symmetric key K

p 89 property 4.2: Comment: The pseudo-random number generators and hash functions have similar features. In [TW06], some good pseudo-random number generators are also used as one-way hash functions. For better presentation in the thesis, we chose the one-way hash function for its formality in describing the dynamic key generation.

p 89 line 21: After the first paragraph of section 4.1.3, insert the following paragraph:

"The following assumptions are valid for the dynamic key generation:

- Alice and Bob share an initial secure channel to exchange EK and IK. The key exchange can be performed via a message encrypted with a shared symmetric key or an asymmetric key exchange

protocol (Diffie Hellman, MQV, Oakley …). The secure channel for key exchange is only used once in the dynamic key generation scheme.

- There is no requirement for the symmetric encryption using encryption key *EK*.
- TK_1, TK_2, …, TK_m and *EK*, *IK* are generated randomly by a secure pseudo-random number generator."

p 91 line 9 equation 4.6: After "...", insert:

$$DK_m = f(SK \oplus TK_m \oplus DK_1 \oplus ... \oplus DK_{m-2} \oplus DK_{m-1})$$

$$DK_{m+1} = f(SK \oplus DK_1 \oplus DK_2 \oplus ... \oplus DK_{m-1} \oplus DK_m)$$

…

p 92 line 5: After "know.", insert: "For every two polynomial algorithms $Al(.)$ and $Al'(.)$"

p 93 line 2: After "therefore", insert: ", for every positive polynomial statement $p(.)$ and for all sufficient large k, the one-way function condition in equation (2.1) is rewritten as"

p 98 line 9: Comment: The multiple membership entities are explained in the user group assignment and the service group assignment relationships in pages 62 and 63. The multiple membership characteristic is also mentioned in the RBAC model [SCFY96]. In reality, an individual user normally can play multiple roles represented by multiple memberships. For example, a user in an online university system has roles as a student of the multiple classes in which s/he has enrolled. In another example, an employee in a company can be a member of the multiple projects in which s/he is involved. One user with multiple memberships is thus a common occurrence. Similar to users, services can also belong to multiple service groups.

p 101 lines 7 and 8: Comment: *GKS* and *CKS* were introduced as group key servers in [WLS07]. Because of the hybrid-based approach, the group key management scheme contains both a centralised group key server and multiple distributed key servers. *GKS* is the centralised key server that manages group membership changes in the entity layer and key-group layer. In the detail layer, a distributed cell key server, *CKS*, controls the membership changes in its cluster only. Therefore, the wireless network system is divided into (z+1) clusters according to (z+1) *CKSs*. Among them, one cluster is the leader-cluster, others are member-clusters.

p 103 line 6: After table 4.2, insert the following paragraph:

"The following are assumed for the group key management rekeying operation:

- Each entity *e* shares an individual sequence of dynamic keys { DK_i^e } with the *GKS*.
- Group keys $K_{KG_i}, K_{SG_j}, K_{EG}$ and $K_{cluster}$ are generated by secure pseudo random number generators.
- The symmetric encryptions in the rekeying operations are infeasible to break (except through brute-force attacks)."

p 119 lines 10-20: Comment: The key management module (working with group key management) in the key management layer produces group keys. These keys become inputs for the authentication module (which controls the authentication protocol in the authentication layer) as authentication keys in the description of the authentication architecture on page 71. In the realisation of the AMUS model in chapter 4, they are specified as K_{EG}^{auth} on page 117. The keys are used to exchange key materials that generate dynamic keys in the authentication protocols described in sections 4.4.1 and 4.4.2.

p 121 line 16: After the end of the paragraph, insert the following paragraph:

"The following assumptions are made for the two authentication protocols:

- User *u* and *AS* share the authentication key K_{UG}^{auth}. Because it is a member of user group *UG*, *AS* is able to obtain the authentication key K_{UG}^{auth}. The key exchanging and sharing are performed in the key management layer.
- Service *s* and *AS* share the authentication key K_{SG}^{auth}. Because it is a member of service group *SG*, *AS* is able to obtain the authentication key K_{SG}^{auth}. The key exchanging and sharing are also done in the key management layer.
- Both *u* and *s* trust *AS* and utilise *AS* as a trusted third party for authentication.
- The symmetric encryptions used in the protocols are infeasible to break (except through brute-force attacks)."

p 145 line 6 theorem 5.2: Comment: In addition to the SVO verification in 5.1.3, all the possible attacks are enumerated and individually analysed in section 5.1.4. Although the ability of the AMUS authentication realisation to resist replay attacks is confirmed by the SVO verification, the analysis in section 5.1.3 is presented in such a way that the resistances of internal, external replay, phishing and cryptanalysis attacks are investigated together. This is to show clearly the security feature of the proposed authentication.

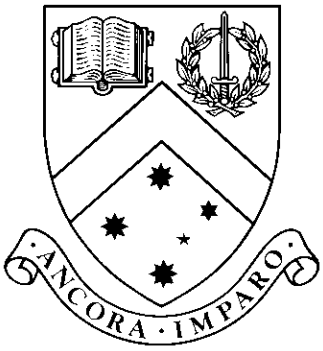
p 179 lines 7 and 14: Comment about equations 5.13 and 5.14: Because the size of the search domain of brute forcing for a dynamic key is 2^s, the probability of correctly guessing an individual dynamic key DK_j is $Pr(Al(.) = DK_j) = \frac{1}{2^s}$. Meanwhile, the probability of breaking the whole sequence of dynamic keys (a set of *n* dynamic keys) is determined in equations 5.13 and 5.14 by the probability of breaking *m* continuous dynamic keys DK_{i-1} to DK_{i-m} and the seed key *SK*. The greater the number of inputs required to generate dynamic keys, the harder it is to correctly guess the whole sequence of dynamic keys (containing multiple dynamic keys).

p183 line 11: Comment: The synchronisation problem in dynamic key cryptography actually does not affect the proposed authentication and its performance. In fact, the authentication protocols presented in pages 121 and 124 only use the first two dynamic keys, DK_1 and DK_2. The first dynamic key DK_1 is used for the authentication of *u* while DK_2 is used for the mutual authentication of *s*. Attacks on the two dynamic keys, DK_1 and DK_2, make either *u* or *s* fail to encrypt or decrypt the two last messages with the correct dynamic keys and terminate the authentication protocols. Hence, there is no need for the dynamic key synchronisation.

Dynamic Group-Based Authentication in Wireless Networks

by

Huy Hoang Ngo



Thesis

Submitted by Huy Hoang Ngo

for fulfillment of the Requirements for the Degree of

Doctor of Philosophy (0190)

Faculty of Information Technology
Monash University

June, 2010

Dynamic Group-Based Authentication in Wireless Networks

Abstract

Authentication is an important part of any computer network. Authenticating individuals and groups of users and services securely and efficiently is challenging, especially in wireless networks. This is because users and services in wireless networks are vulnerable to attack due to the nature of wireless communication and the limitations of wireless devices. The problem is compounded by the need for authentication processes to also be scalable and flexible. Users and services in wireless networks are not only more dynamic than those of wired networks but also greater in number. Authentication processes also need to be able to employ different authentication protocols so that the requirements of different computer networks can be met. A good authentication model for wireless network users and services thus needs to have four desirable properties: *security, efficiency, scalability and flexibility*. Existing authentication models do not sufficiently possess these characteristics. This thesis presents a novel authentication model aimed to achieve these four major properties.

The proposed authentication model consists of *a collection of relationships, a group manager* and *an authentication controller.* In this model, users and services are grouped into user groups and service groups respectively. The collection of relationships of users, services and their groups in this model is defined and classified in order to provide proper authentication for both individuals and groups of users and services. The group manager and the authentication controller are proposed in order to allow authentication with the four desirable properties to be achieved.

In order to demonstrate the practical value of the proposed authentication model, an architecture is derived followed by a realization. The derived architecture has two layers: the key management layer and the authentication layer. Group management and authentication key distribution are conducted in the key management layer while authentication verification is performed in the authentication layer.

We also propose the use of dynamic key technique and group key management in the authentication model. Membership-oriented group key management, adapted for wireless networks, is used to implement the group manager. A dynamic key generation scheme is proposed to create dynamic key sequences and dynamic keys are used to secure communications in both the key management layer and the authentication layer. In order to perform authentication verification, two authentication protocols (ticket-based and request-based) are proposed for the authentication verification of both individuals and groups of users and services and their merits are analysed.

Our analysis and evaluation show that the application of the dynamic key scheme and group key management enable security and efficiency properties for authentication. At the same time, the two layers architecture and the proposed authentication model itself achieve flexibility and scalability.

In summary, the proposed model, along with the derived architecture and its realisation using dynamic key theory and group key management, offers secure, efficient, scalable and flexible authentication for individuals and groups of users and services in dynamic and large wireless networks.

Dynamic Group-Based Authentication in Wireless Networks

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Huy Hoang Ngo
June 16, 2010

This thesis is dedicated to my beloved parents.

Acknowledgments

Thanks to...

... my supervisors Prof. Balasubramaniam Srinivasan, Dr. Phu Dung Le and Dr. Campbell Wilson for their help and guidance.

... my parents Huy Ho Ngo, Phuong Mai Huynh, my uncle The Minh Ngo, and my girlfriend, Thi Hong Nhung Vu, for their love and support.

... my colleague Dr. Xianping Wu and Dr Yi Ling Wang (Tony) for their help, discussion and encouragement.

... my friends Minh Duc Cao, Minh Viet Le, Osama Dandash and Minh Ngoc Nguyen for their support and encouragement.

... Rachel Cawsey and Monash Research Graduate School for their financial support to complete this thesis.

... Dr. Maria Indrawan, Dr. Trent Mifsud, Dr. Flora Dilys Salim, Robert Gray, Duke Fiona and anyone I forgot to mention.

Huy Hoang Ngo

Monash University

June 2010

Publications

Journal Papers

Huy Hoang Ngo, Xianping Wu, Phu Dung Le and Balasubramaniam Srinivasan. An Individual and Group Authentication Model for Large Scale Wireless Network Services. *Journal of Convergence Information Technology*, 5(1):82-94, 2010.

Xianping Wu, Huy Hoang Ngo, Phu Dung Le and Balasubramaniam Srinivasan. Novel Hybrid Group Key Agreement for Sensitive Information Systems. *Journal of Convergence Information Technology*, 5(1):69-81, 2010.

Huy Hoang Ngo, Xianping Wu, Phu Dung Le, Campbell Wilson and Balasubramaniam Srinivasan. Dynamic Key Cryptographic and Applications. *International Journal of Network Security*, 10(3):161-174, 2010.

Xianping Wu, Huy Hoang Ngo, Phu Dung Le and Balasubramaniam Srinivasan. Novel Authentication & Authorization Management for Sensitive Information Privacy Protection using Dynamic Key Based Group Key Management. *International Journal of Computer Science and Applications*, 6(3):57-74, 2009.

Conference Papers

Huy Hoang Ngo, Osama Dandash, Phu Dung Le, Balasubramaniam Srinivasan and Campbell Wilson. Formal Verification of a Secure Mobile Internet Banking Protocol, submitted to *The Second International Conference on Networks & Communications*, Chennai, India, 2010.

Huy Hoang Ngo, Xianping Wu, Phu Dung Le and Balasubramaniam Srinivasan. An Authentication Model for Wireless Network Services - *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications*, pages 996-1003, Perth, Australia, April 2010.

Huy Hoang Ngo, Yiling Wang, Phu Dung Le, Balasubramaniam Srinivasan and Vishv Malhotra. A Membership Oriented Group Key Management for Application Services. *Proceedings of the 12th International Conference on Network-Based Information Systems*, pages 240-245, Indianapolis, USA, August 2009.

Yiling Wang, Huy Hoang Ngo, Phu Dung Le, Balasubramaniam Srinivasan and Vishv Malhotra. Multicasting Key Management in Wireless Networks. *Proceedings of the 12th International Conference on Network-Based Information Systems*, pages 234-239, Indianapolis, USA, August 2009.

Huy Hoang Ngo, Xianping Wu, Phu Dung Le and Campbell Wilson. Package-Role Based Authorization Control Model for Wireless Network Services. *Proceedings of the 4th International Conference on Availability, Reliability and Security*, pages 475-480, Fukuoka, Japan, March 2009.

Xianping Wu, Huy Hoang Ngo, Phu Dung Le and Balasubramaniam Srinivasan. Design & Implementation of a Secure Sensitive Information System for Wireless Mobile Devices. *Proceedings of Australasian Telecommunications Networks and Applications Conference*, pages 45-50, Adelaide, Australia, December 2008.

Xianping Wu, Huy Hoang Ngo, Phu Dung Le and Balasubramaniam Srinivasan. A Novel Authentication Protocol for Sensitive Information Privacy Protection Using Dynamic key based Group Key Management. *Proceedings of The First International Workshop on Network Security & Applications*, pages 1113-1119, Pusan, South Korea, November 2008.

Huy Hoang Ngo, Xianping Wu, Phu Dung Le and Campbell Wilson. A method for Authentication services in wireless networks. *Proceedings of the 14th Americas Conference on Information Systems*, pages 1-9, Toronto, Canada, August 2008.

Xianping Wu, Huy Hoang Ngo, Phu Dung Le and Balasubramaniam Srinivasan. A Novel Group Key Management Scheme for Privacy Protection Sensitive Information Systems. *Proceeding of the 2008 International Conference on Security and Management*, pages 93-99, Las Vegas, United States, July 2008.

Huy Hoang Ngo, Xianping Wu and Phu Dung Le. A Group Authentication Model for Wireless Network Services based on Group Key Management. *Proceedings of the Tenth International Conference on Enterprise Information Systems*, pages 182-188, Barcelona, Spain, June 2008.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Providing strong authentication for wireless network users and services, whether for individuals or for groups, is both important and difficult. It is important because, of all the factors influencing the adoption and use of wireless networks for business, security is the greatest concern [Bry04]. It is also important because of the large number of people and organisations nowadays using mobile devices and wireless networks. During the last few years, with the trend of hosting online applications as services in cloud computing [RR09], the number of services has rapidly increased. With data from individuals and companies being processed and stored online, wireless services are being used more frequently. Yet the characteristics of wireless networks (that is, the physical nature of broadcasting signals in communications, the resource restraints of mobile devices and the large-scale and dynamic nature of users and services) render users and services vulnerable to attack. Providing strong authentication is difficult because the same characteristics make the provision of appropriate security problematic. This chapter explains these issues and key terms, introduces the requirements of a solution and describes the structure of the thesis.

1.1 Wireless Network Users and Services

In wireless networks, the term "users" usually refers to humans using mobile devices to access services via wireless networks, while the term "services" refers to processes receiving requests from users via wireless networks to perform specific tasks. With wireless networks enabling mobile devices, users are able to access services from anywhere and at any time. With the rapid growth of wireless communication techniques and mobile devices, interest in developing wireless network applications and services has also quickly developed [Sal04] and an increasing number of networks are now providing wireless access and services to mobile users [RSA06].

Services can be found in many different forms, especially in wireless networks. They can be found, at the application layer, [RBK08] in communication and messaging services, entertainment services, transaction services and business services. Furthermore, wireless network services can be run on dedicated computers which can either be wired or wireless (such as a web service) or on low-profile mobile devices (such as a peer-to-peer service). Many applications in wireless networks can therefore be seen as services. Cloud computing [DPK+09] (the Software-as-a-Service model (SaaS) [VRMCL09] [BLB+00]) and Service Software Architecture (SOA) [Bel08] are both examples of modeling applications as services.

In regard to structure, users and services can be organised as individuals or groups. Although individual authentication is a common concept, group identities are rarely used in authentication. However, creating group structures is an approach commonly used to reduce the scalability problem of large systems. In distributed systems, grouping users, services or data into sub-groups called domains to be processed locally enhances scalability and availability. Groups of users and groups of services have become familiar in group collaboration and communication applications (for example, Google WAVE [Par10]) and cloud computing services (Google Apps [Cra09]). Group

communication techniques can also utilise group structures to improve communication among group members in wireless networks.

1.2 The Characteristics of Wireless Network Users and Services

1.2.1 Dynamism and Scalability

In wireless networks, users and services are more dynamic and greater in number than wired networks. Once free from wires, users and services become dynamic because mobile devices, via wireless networks, enable 'anywhere anytime' computing. This mode of computing facilitates user access, creating changes in the volume of active users and services. In addition, with seamless connections and great numbers of wireless network devices, the number of overall users and services are higher than in wired networks. Because of the ability of roaming between different base stations in wireless networks, mobile devices are no longer bound to fixed network IP addresses. Furthermore, some services, such as mobile agents, are able to migrate among different devices. These features all contribute to the convenience, and hence popularity, of mobile computing. However, alongside the convenience of wireless networks, users and services also experience challenges. Adelstein et. al. [AGIS05] classify the challenges of wireless networks into three categories: resources, mobility and security.

1.2.2 Resource Issues

Mobile devices, because of their size, have limited bandwidth, limited computational and storage capacity and limited battery power. Yet mobile devices, unlike regular computers, depend on battery power. The more workload the processors have to process, the more battery power they consume. Communications in wireless networks

usually have low bandwidth, high latency, high error rates and low throughput. Because of the above constraints, applications and services in wireless networks have to be very efficient in computation, communication and storage.

1.2.3 Mobility Issues

The second challenge of mobile devices in wireless networks relates to changes in location and network signals. While moving, the network bandwidth can vary rapidly depending on the wireless network signals at the current location. In addition, wireless network users and services may experience frequent loss of connections and handoff [HZS03]. The accumulation of overheads from frequent handoff operations may cause serious performance concerns.

1.2.4 Security Issues

Communications in wireless network services are more vulnerable than in wired networks. Any adversary with a transceiver is able to capture, replay and even modify communication messages in wireless networks. Services and users in wireless networks therefore become more susceptible to security attacks. Sharma and Nakaruma [SN03] summarise security risks in wireless networks as unauthorised access, interception, virus, identity theft, man in the middle attacks (replay attacks; session hijacking), eavesdropping, denial of service attacks and cryptanalysis attacks. These security attack risks (except denial of service attacks) aim to obtain unauthorised access to either sensitive data or services.

One of the main approaches to securing communications and protecting wireless network communications from the above risks is cryptography. Encryption, combined with integrity code, can protect communication data from being accessed by unauthorised parties. As computers become faster, cryptographic key sizes may increase. However, the cost associated with cryptography using large key sizes is a major hurdle

for mobile devices in wireless networks. Because mobile devices usually have small storage space and limited computational powers, increasing key sizes to strengthen the cryptography may involve higher costs.

Another approach to preventing unauthorised access to users and services is the use of authentication. Unauthorised access is the use of services without permission. To protect services from unauthorised access, authentication, combined with authorisation control, is used to validate access requests. Authentication is used to verify claimed identity, prevent fraud and support access controls to validate authorisation and other security components for non-repudiation.

1.3 The Wireless Network Authentication Problem

The wireless network authentication problem is similar to the traditional authentication problem. However, due to the characteristics of wireless users and services, the authentication problem in wireless networks is more challenging. Before examining the authentication problem in wireless networks, we will now briefly review the traditional authentication problem.

1.3.1 The Authentication Problem

In a traditional definition of authentication [Opp96], authentication is a process in which involved entities try to prove their identities. Normally, entities share secrets with trusted servers, known as authentication servers. By proving the ownership of the shared secrets, trust, represented by a secure communication channel, is created between the entities and requested services. In authentication via networks, the shared secrets are typically used to create cryptographic keys (also known as authentication keys) that encrypt communications between entities and authentication servers.

The authentication keys are supposed to be known only to the entities and the authentication servers. Other parties cannot encrypt and decrypt data unless they know the correlating authentication keys. Through the use of authentication keys, entities can prove their identities by decrypting and encrypting messages.

The shared secrets (or factors) can be classified into four types:

- "things that you know" such as a password, a PIN code or pass phrases;
- "things that you have" such as an ID card, a smart card, a SIM card or a token;
- "things that you are" such as a fingerprint, a DNA, or an eye's retina pattern; or
- "things that you do" such as a voice or a written signature.

In practice, the most commonly used factor in authentication systems is "things that you know". Most of the authentication systems use passwords or hash values of passwords as shared secrets [WL92]. More recently, highly secure authentication systems (such as internet banking authentication services) prefer to use a combination of multiple factors for authentication.

In this thesis, we focus on authentication using "things that you know" as the shared secret. Services cannot possess shared secrets related to human authentication such as fingerprints, eye retina patterns and smart cards; instead, authentication for services can only use authentication keys or digital signatures derived from "things that you know". So while authentication for users can utilise any of the above four factors, because an authentication process is required to support both users and services, in this thesis, only authentication methods using "things that you know" as authentication keys are discussed.

Authentication is just one of the major security components used to protect confidentiality, system integrity and non-repudiation. Authorisation also plays a key role in protecting services from unauthorised access. While authentication is used to verify

the identity of users and services, authorisation control is used to validate permission granted to users so they can then access services. When the identity of a request is verified by authentication, access can be granted based on the verified identity and the permission levels of the identity through authorisation control. Consequently, only authorised users can then access services.

1.3.2 The Authentication Problem in Wireless Networks

Traditional approaches for authentication in wired networks experience many issues in wireless networks [BCEP04] [CC05] [CJ03]. Security threats in existing authentication methods (such as replay attacks, man in the middle attacks and cryptanalysis attacks via eavesdropping) are magnified because of the vulnerabilities of wireless network communications. However, due to the resource constraints associated with mobile devices, increasing the cryptographic key size to strengthen security for authentication decreases the performance of mobile devices. Moreover, due to their mobility, wireless network users and services are not only more dynamic but also more numerous than wired users. Larger numbers of users, services and authentication requests result in higher costs in terms of storage, management, searching and verification. These costs may adversely affect the performance of authentication systems. Finally, because mobile devices and service requirements vary widely, flexibility is necessary in authentication processes in order to adapt to the different problems and devices in wireless networks. For all these reasons, providing strong authentication within wireless networks is challenging.

1.3.3 Solution Criteria to the Authentication Problem in Wireless Networks

In order to counter the problems previously described, a strong authentication process for users and services in wireless networks should be able to support dynamic and

numerous users and services to achieve high levels of security, efficiency, flexibility and scalability. These requirements are described as follows:

- **Security**

  Authentication in wireless networks must secure wireless network services from unauthorised access. It also needs to protect wireless network users from identity theft. In other words, the authentication needs to be able to resist attacks in order to provide strong authentication for both wireless network users and services.

- **Performance**

  Authentication in wireless networks must provide operational efficiency in terms of communication, computation and storage to overcome the restrictions of both the wireless networks and mobile devices. It is difficult to provide strong, secure and efficient authentication. Normally strong and secure authentication methods suffer from low performance. In contrast, highly efficient authentication methods have security issues. Achieve both security and efficiency for authentication in wireless networks is challenging.

- **Flexibility**

  Authentication in wireless networks must achieve flexibility in order to adapt to the many different profiles of mobile devices. Because each service has different security requirements, authentication also needs to be flexible to suit the different security requirements of services.

- **Scalability**

  Authentication in wireless networks must be able to support both small and large-scale wireless network users and services. It also needs to be able to adapt to rapid change in the number of users and services.

1.4 Research Objectives

Providing strong authentication through which mobile users can access services (that is, most applications), either for individuals or for groups, is not only important but also difficult. Of the authentication criteria, security is the most important objective and is also the hardest to achieve. This thesis proposes a secure, efficient and flexible authentication method for large-scale and dynamic wireless network users and services. We achieve this by comparing different authentication methods to determine an appropriate solution for authenticating group and individual users and services. The primary objectives of this thesis include:

- proposing a new wireless network authentication model to support dynamic and large-scale users and services;
- developing strong authentication mechanisms and schemes to achieve the security and other goals of the proposed authentication model; and
- formally verifying the correctness of the proposed authentication model through a realisation of the model.

1.5 Contributions of the Thesis

This thesis makes a number of contributions:

- The proposal of a formal and unified authentication model for wireless network services and mobile users.

  In this authentication model, components, and the relationships between components, are identified. These building blocks enable the design of a model that is able to provide secure, efficient, flexible and scalable authentication for wireless network services and mobile users. Based on the basic components of this

model, an authentication architecture is derived as a guideline to implement authentication solutions.

- The development of a dynamic key mechanism for wireless network authentication in order to mitigate some common attacks.

  This dynamic key mechanism is used to secure authentication protocols in the proposed authentication model. Based on the ability to resist replay attacks and cryptanalysis attacks, the dynamic key cryptography can support strong authentication. The efficiency of the dynamic key scheme also has implications for low-resource mobile devices and wireless networks.

- The development of a group key management scheme to support secure and efficient authentication.

  The group key management scheme is a key management scheme to support group authentication in the proposed realisation. The proposed group key management scheme supports multiple group membership users and services so that it can manage and distribute authentication keys to authorised users and services. In addition to performing key management, the scheme is also used to support services using secure group communications.

- Construction and verification of a realisation of the proposed authentication model for practical use of the research work.

  The realisation is developed from the proposed authentication architecture based on the cellular wireless networks. The realisation is used to demonstrate the model's ability to provide both security and efficiency for wireless network users and services. The analysis of the realisation shows that it can achieve security without sacrificing performance.

1.6 Thesis Organisation

The thesis is organised as follows:

Chapter 2 reviews existing authentication methods and related work. The advantages and disadvantages of these existing authentication methods are investigated in detail in this chapter.

Chapter 3 presents a novel authentication model for wireless network users and services and an authentication architecture. An authentication architecture is developed from the authentication model to be a guideline to build authentication implementations for the model. A discussion of the scalability and flexibility of the authentication model is also conducted in this chapter.

Chapter 4 describes a realisation of the authentication model based on the proposed authentication architecture. A dynamic key cryptography mechanism and a group key management scheme are also proposed in this chapter. These are used to support components of the authentication architecture.

Chapter 5 analyses and discusses the security and efficiency of the realisation described in the previous chapter. In this chapter, the security is formally verified from each component to the entire realisation of the model. The communication cost and computational cost of each component, and the relationships among these costs, are also examined to determine the total cost for the authentication of the realisation. The results of the analysis for the proposed realisation are compared with existing authentication methods to validate the correctness of the authentication realisation and the model.

Chapter 6 summarises the research work of this thesis and highlights the contributions. Possible future research is also discussed in this chapter.

Chapter 2

Background and Related Works

This chapter reviews existing approaches, issues and concepts relating to authentication for wired and wireless networks. Because modern authentication approaches in wired and wireless networks rely heavily on cryptography, a brief review of cryptography and key management is presented. Existing authentication approaches are then reviewed and evaluated using the criteria of security, efficiency, flexibility and scalability. Common attacks on authentication are described in order to evaluate the security of the existing authentication approaches. A formal method, SVO Logic, is also introduced. This method is used in a later chapter to formally verify the security of the proposed authentication protocol. We conclude that, as no existing authentication approach meets the four criteria, a new authentication protocol is required that is secure, efficient, flexible and scalable, and is suitable for both individual and group wireless network users and services.

2.1 Cryptography

Cryptography is a crucial mechanism in any security system, but it is especially relevant to authentication. From a security aspect, data integrity and confidentiality are

also vital. Confidentiality is concerned with resources being accessed only by authorised users while integrity refers to protection against unauthorised modification. Integrity and confidentiality are often related to authentication and authorisation. Cryptography can be used to protect data from unauthorised access; in fact, network authentication utilises cryptography to authenticate users and services. Thus cryptography plays a crucial part in modern authentication.

There are two basic types of cryptosystems [Sch96]: symmetric and asymmetric. In symmetric cryptography, the keys used for encryption are the same as those used for decryption. In contrast, asymmetric cryptography uses different encryption and decryption keys. These two types of cryptosystems are described in the following sections along with two other types of cryptosystems: the one-way hash function and the one-time pad.

2.1.1 Symmetric Cryptography

Symmetric cryptography refers to encryption methods that require both the encrypting and decrypting parties to share a secret key. In other words, the keys used in encryption and decryption are the same. Symmetric cryptography is often classified into two common methods: block ciphers and stream ciphers. In block ciphers, data are encrypted and decrypted in blocks of the same size. Block cipher algorithms include the Data Encryption Standard (DES) [Sch96] and the Advanced Encryption Standard (AES) [DR02]. In contrast to block ciphers, stream ciphers create a long stream as a key to be combined with plain text for encryption and decryption. The stream cipher most commonly used is RC4 [PP03].

Among its disadvantages, symmetric cryptography has two major weaknesses: key management and key exchange problems. In communication using cryptography, both the sender and the receiver must share a secret key. The more parties a user communicates with, the higher the number of shared keys that have to be managed.

Key management consequently becomes more complicated. If there is no pre-defined key, a secure key exchange is required. Public key cryptography techniques are used to overcome these disadvantages.

2.1.2 Asymmetric Cryptography

Asymmetric cryptography uses two different keys: one for encryption and the other for decryption. The key for encryption - the ”public key” - is assumed to be well known. In contrast, the key for decryption is kept secret and therefore called the ”private key”. This pair of keys is generated from a complex mathematical problem so that the private key cannot be computed from the public key. The two most common asymmetric cryptography techniques are RSA [MvOV96] and elliptic curve cryptography [Kob87].

2.1.3 An Evaluation of Symmetric and Asymmetric Cryptography

Each type of cryptography has its own advantages and disadvantages. Because of its characteristics, asymmetric cryptography is more secure than symmetric cryptography in key distribution and exchange. However, symmetric cryptography is faster than asymmetric cryptography. According to a benchmark of Crypto++ 5.6 [Dai09] (a library-implemented RSA and AES), an encryption AES/ECB 128 bit key takes 0.14 microseconds while a RSA 1024 bit key takes 0.08 milliseconds for a single encryption operation. Furthermore, Blaze [BDR+96] states that the size of an asymmetric cryptography key must be ten times or more than that of a symmetric cryptography key in order to have a similar level of security. In 2003, RSA Security [Kal03] claimed that an RSA 1024 bit key size cryptography was as secure as an 80-bit symmetric key cryptography and that 2048-bit RSA keys were equivalent to 112-bit symmetric keys. Although increasing key size can enhance the security of the cryptography, it also degrades performance. Because the disadvantages of symmetric cryptography are the

advantages of asymmetric cryptography (and vice versa), symmetric and asymmetric cryptographies are often combined together to secure cryptographic protocols.

Asymmetric cryptography is commonly used to exchange session keys while communications are secured by symmetric cryptography using session keys. In SSL [Bar06] / TSL [MSS98], asymmetric cryptography (such as Diffie-Hellman [DH76] and elliptic curve cryptography) [HMV04] operates the key exchange between clients and servers to distribute session keys. After that, session keys are used as symmetric cryptographic keys for encrypting all messages in the one communication session. Each session key can only be used within one session.

2.1.4 The One-Way Hash Function

The one-way hash function is special type of cryptography. In [TW06], a one-way function $f(.)$ is defined as a mathematical function that is easy to compute but much harder to invert. In other words, an algorithm takes polynomial time to compute function $f(.)$. However, there is no probabilistic algorithm to compute the inverse function $f^{-1}(.)$ in polynomial time. Normally, a hash function is used to compute a signature to authenticate the source of a message. In mathematics, the definition of a one-way hash function contains three parts:

i. $f(x)$ is a polynomial function (so that it is easy to compute $f(x)$ from x);

ii. a polynomial algorithm Al exists so that$\forall x, Al(x) = f(x)$; and

iii. for every probabilistic polynomial time algorithm Al', every positive polynomial statement $p(.)$ and for all sufficiently large k:

$$Pr(Al'(f(x), 1^k) \in f^{-1}(f(x)) < \frac{1}{p(k)} \tag{2.1}$$

Damgard [Dam90] added a collision-free condition for the one-way hash function and renamed it a strong one-way hash function. When Ag is any algorithm, to find

two distinct messages, M and M', every positive polynomial statement $p(.)$ and all sufficiently large k, the collision-free condition for one-way hash functions is written as follows:

$$Pr(Ag(M, 1^k) = M', M \neq M', f(M) = f(M')) < \frac{1}{p(k)} \tag{2.2}$$

There are several well-known one-way functions:

- one-way functions based on computational number theory (for example, RSA [RSA78], Rabin [MvOV96] and Discrete Logarithms [Sti05]);
- trapdoor permutation functions (such as RSA Trapdoor [DH76] and Clawfree Permutations [Gol07]); and
- message hash functions [Sch96] (such as MD2, MD5, SHA and HAVAL [ZPS93]).

2.1.5 The One-Time Pad

The one-time pad [Kah67] is a symmetric key cryptography using random cryptographic keys called pads that are only used once. A pad is derived from a random stream of numbers. When the pad is perfectly random and the size of the pad is the same as the input plain text, the cryptography can be proved to possess perfect secrecy [Sha49].

Despite having the property of perfect secrecy, the one-time pad has the following major obstacles in practice:

- perfectly random pads are hard to achieve;
- the pads require secure generating and distributing among parties; and
- no attempt is made to to ensure the pad is used only once.

Because of the implementation difficulty of securely generating and exchanging the pad, it is not feasible to use one-time pad cryptography to ensure secure communication via networks.

This section reviewed different approaches to cryptography. The following section describes a key management scheme. The scheme manages and distributes symmetric cryptography keys used to secure group communications. These symmetric keys can also be used as authentication keys for group authentication.

2.2 Group Key Management

2.2.1 Group Key Management and the Secure Group Communication Problem

The emergence of the group communication model has enabled the development of collaborative applications that support functions such as emergency response, on-line meetings and on-line live auctions. These applications can use multi-cast addressing to achieve cost effectiveness in communication between group members. With wireless networks, these applications can also utilise the natural broadcasting signal in wireless networks to extend the efficiency of multi-cast in communications. However, multi-cast communications are more vulnerable than traditional uni-cast communications. Securing group communications by assuring integrity and confidentiality is critical.

Among the available solutions to secure group communication, a low-cost approach is to employ group key management [CS05] to manage and distribute symmetric keys among group members. These symmetric keys (also called group keys) are used to secure communications between group members.

Where $\mathbb{U}$ denotes a set of users, $\mathbb{K}$ a set of keys and $\mathbb{R} \subset \mathbb{U} \times \mathbb{K}$ denotes a user-key relationship, group key management is formalised as follows:

$$GKM = \{\mathbb{U}, \mathbb{K}, \mathbb{R}\} \tag{2.3}$$

There are five security requirements [CS05] for group key management: forward secrecy, backward secrecy, collusion resistance, key independence and minimal trust. Among these, Zou et. al. [ZRM05] state that *forward secrecy* and *backward secrecy* are the most important requirements. *Forward secrecy* ensures that entities that have left the group cannot obtain any future keys. *Backward secrecy* ensures that entities that have joined the group cannot obtain any previous keys. To ensure *forward secrecy* and *backward secrecy* in group key management, rekeying operations must be performed after each member joins or leaves the group. In large and dynamic groups, the cost of rekeying operations becomes a major performance concern because of the high numbers of multi-cast messages and encryptions necessary to update group keys.

Different approaches have been proposed with the aim of reducing the cost of rekeying operations. In centralised approaches [PACB98] [CQN02], a server called a key controller (KC) is dedicated to handling the rekeying operations of group key management. In contrast, distributed approaches [CCSI01] [Opp96] use several KCs to manage rekeying operations. However, in large and dynamic groups, join and leave requests can be made frequently and these group key management schemes, despite their laudable aims, cannot perform rekeying operations efficiently in large and changeable groups. In an effort to reduce the computational cost in these groups, Wong et. al. [WGL98] proposed a centralised scheme known as the Logical Key Hierarchy (LKH). LKH utilises a tree of auxiliary keys to reduce the cost of the rekeying operation resulting from a single member leaving a group. To further reduce the rekeying costs associated with multiple join and leave requests, Li et al. [LYGL01] presented a method to batch rekeying requests. Although this batching operation

improves rekeying performance [ZMBL04] [CCE05] [CCE08], it postpones the rekeying resulting from simultaneous member join and member leave operations. Consequently, during the batching period, evicted group members can still access group keys, thus allowing unauthorised access. Li et al.'s approach [LYGL01] is thus not suitable for use with authentication. Instead, our following discussion reviews the LKH structure as a standard key management scheme.

2.2.2 Logical Key Hierarchical Structure

In an LKH structure [WGL98], the KC manages a secure group with a tree graph structure. To manage group members efficiently, a tree structure of group keys is defined that arranges group members in sub-groups. Each node of the tree, according to the sub-group, contains a secret group key shared among the group members. The highest node of the tree is called the root and stores the key shared by all members in the system. The nodes without child nodes are known as leaf nodes. Each user has an individual secret key and all the group keys according to its group from the leaf node to the root.

Figure 2.1 shows the tree structure of an LKH secure group that has eight members: $u_1, u_2, u_3, u_4, u_5, u_6, u_7$ and u_8.

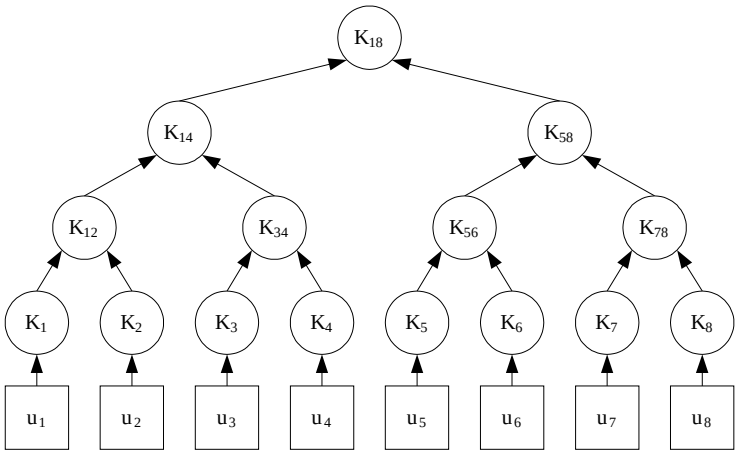


Figure 2.1: A Tree Structure of the Logical Key Hierarchy

The tree structure is specified by three parameters:

- the number of users n;
- the height h that is defined as the length of the longest path from the root to the leaf node; and
- the degree d that is the maximum number of child nodes from any node in the tree.

If the tree is a balanced tree, we can say that $h = log_d n$. These parameters are used to determine the performance of re-keying operations. In the hierarchical structure example in figure 2.1, n is 8, h is 3 and d is 2. As mentioned above, rekeying operations are applied to protect forward secrecy and backward secrecy for the group key.

2.2.3 Rekeying Operations

Notation

The notation used to describe the rekeying protocols are listed in table 2.1.

Table 2.1: LKH Notation

$u_1, \ldots, u_8$	users of the group keys.
$P \rightarrow Q : X$	P sends a message X to Q via uni-casting.
$P \Rightarrow everyone : X$	P sends a message X to all members in the group using multi-casting.
K_Y	group key of group Y.

This notation is used to describe the rekeying operations. There are two major rekeying operations: rekeying for a member join operation (henceforth abbreviated to "member join") and rekeying for a member leave operation (abbreviated to "member leave").

Member Join Rekeying

To describe the rekeying for member join, the example illustrated in figure 2.1 is used. Assume that the key structure is started with seven members ($u_1, u_2, u_3, u_5, u_6, u_7$ and u_8), and user u_4 wants to join the group. u_4 sends a join request to the KC. After being authenticated, u_4 is authorised to join. The group keys K_{34}, K_{14} and K_{18} are updated by rekeying in preparation for the join request of user u_4. The rekeying messages are described as follows:

1. $u_4 \rightarrow KC$: {join request}.
2. $KC \Rightarrow$ everyone : $\{K'_{18}\}K_{18}, \{K'_{14}\}K_{14}, \{K'_{34}\}K_{34}$.
3. $KC \rightarrow u_4 : \{K'_{18}, K'_{14}, K'_{34}\}K_{u_4}$.

The cost of the rekeying for a join request on the server side is h encryptions and one multi-cast message of h keys. The cost for sending the new group keys to new member u_4 is also h encryptions and one uni-cast message of h keys.

Member Leave Rekeying

The leaving of u_4 in figure 2.1 can also be used to illustrate the rekeying for member leave. Assume that u_4 wants to leave the group. The leave request is sent to the KC. The group keys K_{34}, K_{14} and K_{18} are updated by rekeying for member leave. The update process starts at the leaf node (u_3) and travels up to the root. The rekeying messages are described as follows:

1. $u_4 \rightarrow KC$: {leave request}.
2. $KC \rightarrow u_3 : \{K'_{34}\}K_3$.
3. $KC \Rightarrow everyone : \{K'_{14}\}K_{12}, \{K'_{14}\}K'_{34}$.
4. $KC \Rightarrow everyone : \{K'_{18}\}K'_{14}, \{K'_{18}\}K_{48}$.

The rekeying process is taken from the bottom to the top, excluding the root. Where the height of the tree is h and the degree is d, an individual rekeying for a leave request takes $d \times (h-1) + d - 1$ encryptions. In a multi-cast message, rekeying takes $d-1$ uni-cast message and $h-1$ multi-cast messages with d keys.

Wang et al. [WLS07] point out that LKH structures in large and dynamic wireless networks have performance issues. Although logical group neighbour members are not always in the same physical wireless network, rekeying for a membership change in an LKH structure affects all logical neighbour members in the groups. In addition, in large wireless networks, rekeying for members in different physical wireless networks is an inefficient operation. Furthermore, due to LKH's centralised nature, a large group may cause high storage costs for group keys in the key tree of LKH. To address these issues, Wang et al. [WLS07] proposed a hybrid group key management approach for wireless networks.

2.2.4 Hybrid Group Key Management

Hybrid group key management [WLS07] is a distributed approach to group key management. Based on the structure of wireless cellular networks, the membership management of hybrid group keys is distributed to multiple local wireless network cells. The distributed architecture of the hybrid group key management logical structure is described via two different levels: the fundamental wireless network level and the logical structure level.

In the fundamental wireless network level, the whole wireless network is separated into smaller local wireless cells. Each wireless cell is managed by a key server named CKS that is integrated with the base station of the wireless cell. The CKS locally handles group key operations for members in its wireless cell. A central key server named GKS on the top level manages the messages of CKSs. This structure is illustrated in figure 2.2.

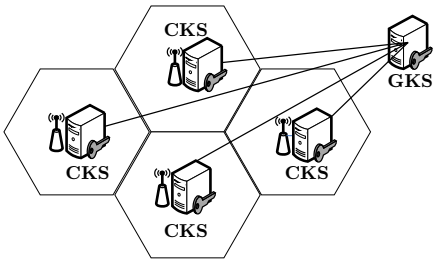


Figure 2.2: The Cellular Network Structure of the Hybrid Group Key Management

Based on the fundamental wireless network structure, the logical structure of the hybrid group key is also divided into two clusters: the leader cluster and the member cluster. Thus the logical structure is separated into two different layers. The leader cluster is in the top layer while the member clusters are in the bottom layer. With these two clusters, rekeying operations in group key management occur independently within each layer. The logic structure is illustrated in figure 2.3.

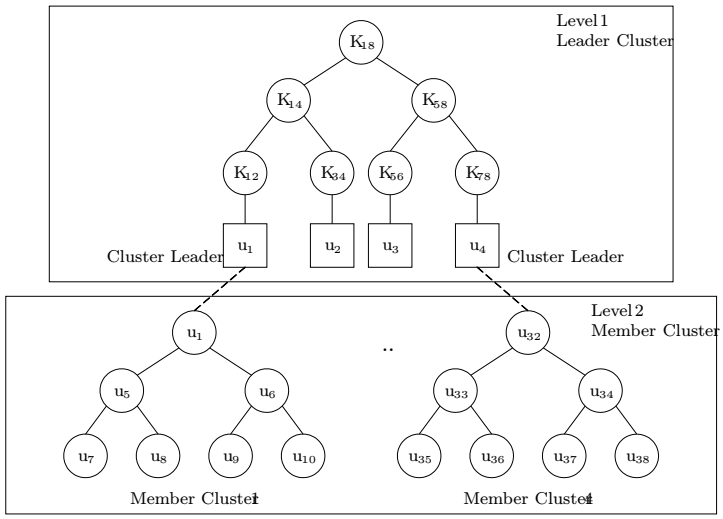


Figure 2.3: The Logical Structure of the Group Key Management

The distributed architecture of the hybrid group key management approach can overcome the performance issues associated with the LKH scheme. By separating the leader and member clusters, the effects of membership changes are localised within their clusters and the costs associated with rekeying operations are reduced. Furthermore, membership management of the hybrid group key is distributed to one GKS

and multiple CKSs in the structure. The costs of storage, computation and communication of rekeying are thus further minimised.

Having briefly reviewed cryptography and key management, we will examine existing authentication approaches. However, before reviewing existing authentication methods, a set of criteria is described. These are the criteria against which existing authentication methods are evaluated. In addition, they also serve as guidelines to help build the proposed authentication model.

2.3 Authentication Evaluation Criteria

As said previously, there are four major criteria for good authentication: (i) security; (ii) efficiency; (iii) flexibility; and (iv) scalability.

2.3.1 Security

Effective security possesses the characteristics of confidentiality and integrity, privacy and attack resistance. Confidentiality and integrity help to protect users and services from unauthorised access. The characteristic of privacy ensures that users are protected from phishing or cryptanalysis attacks that might expose authentication data. Attack resistance refers to the ability to block possible assailment. Attack resistance is discussed in more detail in section 2.4.

2.3.2 Efficiency

Efficiency within an authentication model is a measure of the costs required to manage computation, communication and storage. Computational costs in authentication include costs from encryption, decryption, searching, key generation and verification operations. The communication cost is the cost related to sending messages between

the involved principals during the authentication process via both multi-cast and uni-cast messages. The storage cost results from the number of authentication keys and other materials required to verify authentication.

Behind security, efficiency is the second most important criteria in authentication. Because mobile devices in wireless networks have limited resources, they often have a low-performance processor, small communication bandwidth, small memory and limited battery power. As all operations on mobile devices consume battery power, efficiency in authentication not only reduces the drain on the processing resource and communication bandwidth, but also saves battery power.

2.3.3 Flexibility

Flexibility within an authentication model describes the ability to apply different authentication protocols and cryptographic systems. Because each mobile device may support a different set of authentication protocols and cryptographic systems, a fixed authentication protocol and cryptographic system is not suitable for mobile devices in wireless networks. Furthermore, a single authentication protocol and cryptographic system is also unsuitable where situations require different security levels.

2.3.4 Scalability

Scalability in authentication refers to the ability to adapt from small to large (and vice versa) wireless networks and the capacity to support heavy authentication loads. Bondi [Bon00] classifies scalability into two aspects: structural scalability and load scalability.

- Structural scalability: Because of the dynamism and scalability characteristics of users and services in wireless networks (see section 1.2.1), the number of user and service groups may grow and shrink rapidly. The authentication must be

able to support rapid growth in the number of users and services without having major changes in its architecture. This requires the process to have low storage and management costs, and a low authentication cost. The low authentication cost requirement is identical to the efficiency criteria discussed in 2.3.2. It is therefore omitted in the evaluation for structural scalability.

- Load scalability: Because of their dynamism and the mobility issues, users and services may more frequently join and leave networks and more frequently experience handoff in their wireless networks. The authentication load in wireless networks is thus usually higher than that in wired networks. The authentication must be able to adapt to rapid increases in work load so that authentication throughput can be improved when resources (usually hardware) are added.

Of the above criteria, security is the most important and challenging. It is difficult to achieve strong security in authentication for wireless network users and services using mobile devices with limited computation and communication resources. Communication messages are also more vulnerable to attack. Yet the other criteria (efficiency, flexibility and scalability) are also necessary so that the authentication model can achieve strong authentication for wireless network users and services.

This section has presented the chief requirements of strong authentication. The next section discusses common attacks on authentication methods. These attacks are then used to help evaluate the security of different authentication methods.

2.4 Possible Attacks on Authentication Methods

In this section, possible attacks on authentication methods are described. Three common attacks are discussed in this thesis: replay attacks, cryptanalysis attacks and phishing attacks. While replay attacks try to obtain unauthorised access by exploiting

vulnerability in authentication protocols, cryptanalysis attacks aim to obtain authentication keys or session keys via cracking cryptography. A phishing attack illegally acquires authentication keys and passwords via social engineering or fraudulent authentication services. These attack methods provide a structure against which the security of the existing authentication methods and the proposed authentication model can be evaluated.

2.4.1 Replay attacks

Syverson [SC01] defines replay attacks on cryptographic protocols as efforts of using messages captured from previous or current communications to perform unauthorised operations or to obtain unauthorised access. He describes three common methods of replay attacks: deflection, reflection and straight attacks. The attacks may involve minor modification of replay messages. In [Syv94], replay attacks are classified into two different types: internal replay attacks and external attacks.

In external replay attacks, adversaries who are not involved in the authentication protocol capture authentication messages for future penetration attempts. With sufficient time, it is possible for an adversary to break the session keys being distributed in captured messages. The adversary then tries to persuade participants of authentication protocols to re-use previously compromised session keys. During a penetration attempt, the adversary can replay previously captured messages to deceive a principal into believing that a replayed message is another (valid) key exchange message. Existing authentication protocols vulnerable to this attack include Needham Shroeder [NS78] [DS81], Yahalom [BAN89], and Kerberos [Gon92].

In an internal replay attack, the adversary is a principal involved in authentication. The adversary saves one or more part(s) from a previous authentication message in a protocol for use in a future message. The attacks exploit the similarity in message format between the service ticket request message and the ticket distribution message.

The authentication protocol Neuman and Stubblebine [Syv93] is vulnerable to this attack.

In order to prevent risks from replay attacks, authentication protocols typically use a technique called "challenge-response". In a challenge-response authentication, the principal receiving a request (denoted as Bob) issues a random number (that is, a challenge nonce) and sends it to the principal sending the request (denoted as Alice). In order to authenticate her identity, Alice computes a value, called a response, by encrypting the nonce with her shared authentication key and sending it to Bob. Because Bob's challenge nonce is different in every authentication, replay messages can be detected and the fraudulent authentication request denied. However, the challenge-response technique can be broken by cryptanalysis attacks.

2.4.2 Cryptanalysis attacks

Cryptanalysis attacks [Kah67] aim to compromise secret keys in both symmetric cryptography and asymmetric cryptography by breaking ciphers. According to Schneier [Sch96], most ciphers (except the one-time pad), regardless of the key size, are breakable by brute-force attack if the adversary has sufficient computational resources. By capturing communication messages, adversaries may be able to detect patterns from encrypted messages that increase the chance of breaking the ciphers. With an adequate amount of time, an adversary is able to capture a greater number of challenge nonces and responses and consequently will have a higher chance of breaking the cryptography and correctly guessing the authentication key.

Of the cryptanalysis attack methods, the most commonly used method is brute-force attack which performs an exhaustive search of all possible keys in the key space to find the correct key. The brute-force attack is also the most complex method. If the key size is n bits, the complexity of the brute-force attack is O(2^n). This indicates that the brute-force attack may have to try all possible keys in the key domain to break the

cryptography. Therefore, the smallest chance to break a cryptography using a brute-force attack is $\frac{1}{2^n}$.

Another cryptanalysis attack method, dictionary attack [Wu97], targets authentication systems using authentication keys derived from hashing authentication passwords. Wu argued [Wu97] that the authentication passwords to create authentication keys often have low entropy. The domain of generated authentication keys from a low-entropy password is smaller than the key space. The probability of guessing the correct password to find the right authentication key is therefore higher.

Cryptanalysis attack risks are usually related to re-usable long-term cryptographic keys. According to [Jou09], cryptanalysis attack methods are classified into four types: ciphertext only attacks, known plaintext attacks, chosen plaintext attacks and chosen ciphertext attacks. All of these are based on captured ciphertext and/or plaintext. When a cryptographic algorithm uses one cryptographic key over a period of time, an analysis of captured ciphertext and/or plaintext can find patterns that enable the cryptographic key to be correctly guessed.

To reduce the risk of compromised cryptographic keys from cryptanalysis attack, modern authentication methods use session keys [CC03]. In [Opp96], a session key is defined as a single-use symmetric cryptographic key for encrypting all messages in a communication session. Instead of using a long-term shared key (or an authentication key) as a cryptographic key to encrypt all messages, every session uses a different session key to encrypt messages. The long-term shared key or authentication key is used as a cryptographic key to perform session key exchange. Fewer encryptions therefore use the long-term shared key (which can be either symmetric or asymmetric).

Use of session keys also reduces the security risk because even if a session key is compromised, only communication data within a session will be exposed. However, the session key is often created by key exchange protocols relying on permanent asymmetric keys or symmetric authentication keys. The longer these session keys are used,

the more they expose to cryptanalysis attacks. When these session keys are compromised, the authentication system becomes vulnerable to adversaries. This weakness has been demonstrated many times. For example, in 1995, the long-term cryptography key of the Netscape implementation of SSL 1.0 was broken by Ian Goldberg and David Wagner [VMC02] and successful attempts of cryptanalysis have been made on RSA and SSL/TLS [Bar06] [KPR03] [Koc96] [SHK06]. When a cryptographic key is re-used, even for a limited number of times, an opportunity arises to break the cryptography.

2.4.3 Phishing attacks

Phishing is another method that attempts to criminally and fraudulently obtain authentication keys or passwords. Traditional attacks may come from emails, phone calls or instant messengers asking for usernames and passwords. Attackers can also divert victims to counterfeit websites masquerading as legitimate in order to capture sensitive information such as authentication data or credit cards. In wireless networks, falsified services can be used to acquire authentication passwords or keys by asking victims to authenticate on fraudulent authentication systems.

Because of the lack of mutual authentication, phishing attacks in wireless networks can be more serious. Without being able to identity falsified services, users may reveal their passwords to attackers. The vulnerability is minimised if a mutual authentication is conducted. With mutual authentication, users are able to validate the identity of services in order to detect faked services.

In the next section, a formal method to analyse authentication protocols, SVO (derived from author's names Syverson and van Oorschot), is described. This method is then used to examine whether an authentication protocol is vulnerable under replay attacks or phishing attacks.

2.5 SVO Logic Method to Formalise Authentication Protocols

SVO [SC01] is a widely used technique to verify cryptographic protocols. SVO is a successor of Burrows - Abadi - Needham (BAN) Logic [BAN89], a traditional logic used to verify authentication protocols. A number of extensions [AT91] [GNY90] [vO93] [MB93] have been proposed to overcome the limitations of BAN. In 1993, Syverson and Van Oorschot [SO94] combined these extensions to create SVO.

Among the formal methods to analyse authentication protocols, those incorporating BAN-type logic possess a number of advantages [RH93]. One of the advantages of BAN-type logic is that the goals of the analysed protocols must be specified before the analysis is conducted. The purpose of the protocol analysis using BAN-type logic is to find out whether the goals are achieved or not. Another advantage of BAN-type logic is the simplicity of the notation and logic. According to Rubin and Honeyman [RH93], SVO is a unified logic that inherits notation and semantics from other BAN-type logics.

2.5.1 SVO Logic Notation

Table 2.2 summarises the notation for SVO Logic using symmetric cryptography.

Table 2.2: SVO Notation

P believes X	The principal P acts as if X is true
P received X	The principal P received a message containing X. It can read and replay X.
P said X	The principal P sent a message containing X.
P has X	X is initially available to P, received by P or generated by P.
P controls X	P has jurisdiction over X.
$fresh(X)$	X has not been sent in any messages before the current one.
$P \overset{k}{\longleftrightarrow} Q$	k is a communication shared key between P and Q. k is only known by P, Q and the trusted party (AS).
$\{M\}_k$	Encryption result of message M using key k.
$\langle X \rangle_{*P}$	P is unable to read X (e.g. P receives $\{X\}_k$ and P does not have k) or P does not recognise X.

2.5.2 SVO Rules

SVO has the two following deduction rules:

- **Modus Ponens**: $\varphi \wedge (\varphi \rightarrow \psi)$ infers ψ.
- **Necessitation**: $\vdash \varphi$ infers $\vdash P$ believes φ.

'$\vdash$' is a meta-linguistic symbol. $\Gamma \vdash \varphi$ means that φ is derivable from the set of formulae Γ using the above rules.

2.5.3 SVO Axioms

There are sixteen axioms related to symmetric cryptography authentication protocols.

Belief Axioms

1. (P believes $\varphi \wedge P$ believes $(\varphi \rightarrow \psi)) \rightarrow P$ believes ψ.
2. P believes $\varphi \rightarrow P$ believes (P believes φ).

Source Association Axiom

3. $(P \overset{k}{\longleftrightarrow} Q \wedge R$ received $\{X$ from $Q\}_k) \rightarrow Q$ said $X \wedge Q$ has X.

Receiving Axioms

4. P received $(X_1, \ldots, X_n) \rightarrow P$ received $X_i, \forall i = 1, \ldots, n$.
5. (P received $\{X\}_k \wedge P$ has $k) \rightarrow P$ received X.

Possession Axioms

6. P received $X \rightarrow P$ has X.
7. P has $(X_1, \ldots, X_n) \rightarrow P$ has $X_i, \forall i = 1, \ldots, n$.

8. (P has $X_1 \wedge \ldots \wedge P$ has X_n) $\rightarrow P$ has $F(X_1, \ldots, X_n)$.

   'F' is a meta-notation for any function computable in practice by P.

Comprehension Axioms

9. P believes (P has $F(X)$) $\rightarrow P$ believes (P has X).

   'F' is a meta-notation for any function that is effectively one-one and computable in practice by P.

Saying Axioms

10. P said $(X_1, \ldots, X_n) \rightarrow P$ said $X_i \wedge P$ has $X_i, \forall i = 1, \ldots, n$.

11. P says $(X_1, \ldots, X_n) \rightarrow (P$ said $(X_1, \ldots, X_n) \wedge P$ says $X_i), \forall i = 1, \ldots, n$.

Freshness Axioms

12. $fresh(X_i) \rightarrow fresh(X_1, \ldots, X_n), \forall i = 1, \ldots, n$.

13. $fresh(X_1, \ldots, X_n) \rightarrow fresh\ F(X_1, \ldots, X_n)$.

   'F' is a meta-notation for any function that genuinely depends on all $X_1, \ldots, X_n$.

Jurisdiction and Nonce-Verification Axioms

14. (P controls $\varphi \wedge P$ says $\varphi) \rightarrow \varphi$.

15. $(fresh(X) \wedge P$ said $X) \rightarrow P$ says X.

Symmetric Goodness Axioms

16. $P \overset{k}{\longleftrightarrow} Q \equiv Q \overset{k}{\longleftrightarrow} P$.

2.5.4 SVO Goals

The process of verifying cryptographic protocols using the SVO technique begins with goal setting. Different types of cryptographic protocols have different goals. Once the goals are set, an analysis is then made in an attempt to deduce these goals from original assumptions and the above axioms. When the SVO goals are met, the cryptography protocols are said to be secure. In SVO, authentication protocols have the six following goals:

- **G1. Ping Authentication** demonstrates that a principal P wants to know whether Q is alive. It is expressed by "P believes Q says X".
- **G2. Entity Authentication** demonstrates that a principal P wants to know whether Q says something relevant to the present conversation. Let N_P be fresh information to P. Entity authentication requires Q to have recently sent a message $F(X, N_P)$ that describes that Q has seen N_P and processed it. In other words, entity authentication is expressed as "P believes $fresh(N_P) \wedge P$ believes (Q says $F(X, N_P)$)".
- **G3. Secure Key/Channel Establishment** indicates that a principal P believes that it has a good key k to communicate with Q. It is expressed by "P believes $P \overset{k}{\longleftrightarrow} Q$".
- **G4. Key Freshness** demonstrates that a principal P believes in the freshness of the communication key k in G3. It is expressed by "P believes $fresh(k)$".
- **G5. Mutual Understand of Shared Keys** specifies that P observes that Q has sent a message using k as an unconfirmed secret during the conversation between P and Q. It is expressed by "P believes Q says ($P \overset{k}{\longleftrightarrow} Q$)".
- **G6. Key Confirmation** demonstrates that P believes that Q has received and processed k. It is expressed by "P believes $P \overset{k}{\longleftrightarrow} Q \wedge Q$ says $F(k)$".

The SVO and its six goals are used to analyse protocols in the authentication model in chapter 5. In the next section, existing authentication methods are examined. These methods are revisited in Chapter 5 where they are used to help evaluate the proposed authentication model.

2.6 Existing Authentication Methods

According to Oppliger [Opp96], authentication methods can be classified into two types: authentication with a trusted third party and authentication without a trusted third party. Each type has its own advantages and disadvantages. Authentications using trusted third parties are susceptible to bottlenecks [LC97] [Gon93] and the possibility of a compromised trusted third party [SYS97]. However, without a trusted third party, two parties lacking previous direct mutual trust cannot provide authentication without compromising security [Opp96]. Furthermore, Gong [Gon93] shows evidence of a trade off between increasing security and increasing availability. To counter this problem, he proposed replicating authentication services to remove the bottleneck problem. With a proper mutual authentication protocol, even should a minority of trusted third parties be compromised, the security of the authentication system as a whole would not be jeopardised.

In this section, two authentication methods using trusted third parties (Kerberos and OpenID) and one method without a trusted third party (the Password Authentication Key Exchange Method) are reviewed.

2.6.1 Kerberos Authentication Method

The Kerberos Authentication Method [NT94] [SNS88] uses a trusted third party (a Key Distribution Center (KDC) that includes an authentication server (AS) and a Ticket Granting Server (TGS)) to distribute session keys via authentication tickets.

With these tickets and session keys, users are able to authenticate their identities with services.

The Kerberos authentication protocol has six steps. In the first step, a user sends a request to AS. In the second step, AS issues a ticket granting ticket $Ticket_{c,tgs}$ for the client to authenticate with TGS. In the third step, the client combines a message with the ticket $Ticket_{c,tgs}$ and an authenticator A_u and sends this combination to TGS. After verifying the ticket and the authenticator, TGS creates an authentication ticket $Ticket_{c,s}$ and a session key $K_{c,s}$ and sends these to the user. In turn, the user employs the session keys to encrypt A_u and combine it with $Ticket_{c,s}$ to send to the service. After verifying the authenticator and the ticket, the service expresses trust in the user identity by sending a confirmation that the service is willing to serve the user's requests. The authentication protocol is shown in figure 2.4.

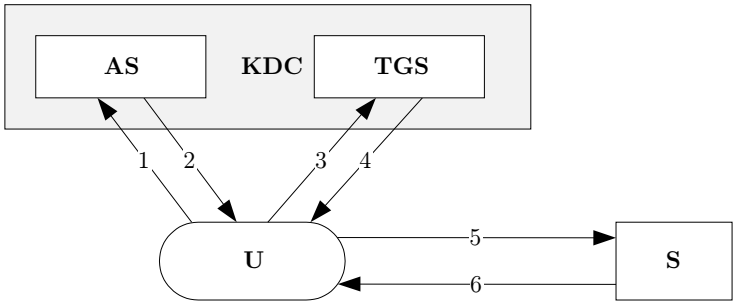


Figure 2.4: The Kerberos Authentication Protocol

Table 2.3: Notation for Kerberos

u	user
s	service
AS	Authentication Server
TGS	Ticket Granted Server
$N_u, N_u^{'}$	nonces
$K_{u,tgs}$	session key shared between U and TGS

$K_{u,s}$	session key shared between U and S
K_u	long-term shared key between U and AS
K_{TGS}	long-term shared key between TGS and AS
T_X, T'_X	timestamps from X
T_{exp}, T'_{exp}	expired timestamps of the tickets
$addr$	network address of U
$Ticket_{u,tgs}$	$\{U, addr, T_{AS}, T_{exp}, K_{u,tgs}\}$
A_u	$\{U, addr, T_U\}$
$Ticket_{u,s}$	$\{U, S, addr, T_{TGS}, T'_{exp}, K_{u,s}\}$

With the notation in table 2.3, the authentication protocol is formalised as follows:

1. $u \rightarrow AS : u, N_u$
2. $AS \rightarrow u : \{K_{u,tgs}, TGS, N_u\}K_u, \{Ticket_{u,tgs}\}K_{TGS}$
3. $u \rightarrow TGS : s, N_u^{'}, \{Ticket_{u,tgs}\}K_{TGS}, \{A_u\}K_{u,tgs}$
4. $TGS \rightarrow u : \{K_{u,s}, N_u^{'}\}'K_{u,tgs}, \{Ticket_{u,s}\}K_{tgs,s}.$
5. $u \rightarrow s : \{Ticket_{u,s}\}K_{tgs,s}, \{A_u\}K_{u,s}.$
6. $s \rightarrow u : \{T_u\}K_{u,s}.$

The Kerberos authentication method has been used for many authentication systems. The original Kerberos method relied on symmetric key encryption for authentication and has formed the basis for most of the standard authentication services in Unix and Windows. Several projects have added public key infrastructure to the Kerberos method. For example, Tung et al. proposed the Public Key Cryptography for Initial Authentication in Kerberos (PKINIT) [TNH+00] and the Public Key Cryptography for Inter-realm Authentication (PKCROSS) [TRN+98]. Medvinsky [MHN97] added Public Key Utilising Tickets for Application Server (PKTAPP). Extensions of the Kerberos Authentication Method have also been made for wireless networks. In the

Charon method, Fox and Gribble [FG96] proposed to add proxies to Kerberos to authenticate on behalf of wireless clients. In another effort to reduce traffic, Pirzada and McDonald [PM04] proposed one-phase authentication in Kaman. These two methods Charon and Kaman are based on symmetric cryptography and timestamps.

2.6.2 OpenID

OpenID [Found] is a single-sign-on solution providing a decentralised authentication service for web users. The original OpenID protocol, Yadis [EHG07], was proposed by Brad Fitzpatrick in 2005. It has been widely accepted by many companies including AOL, Symantec BBC, Google, IBM, Microsoft, MySpace, Paypal, Verisign, and Yahoo [Eld09]. Yadis allows web users to use a single identity from an OpenID identity provider to authenticate with many web application services.

OpenID enables decentralised architectures for authentication. Instead of having one centralised identity provider (such as Microsoft .Net Passport [Opp03]), OpenID supports authentication from different OpenID identity providers. Users are not required to create and maintain new accounts on multiple websites. Instead, they can use a subscribed identity from an OpenID provider to authenticate with any OpenID-supported website.

The OpenID authentication protocol has nine phases:

1. User u establishes a secure channel with s via HTTPS (SSL/TLS) [MSS98]. The process to create a secure channel involves two steps. The first step is key exchange:

   $u \rightarrow s : u, SSL\ Version_u, CryptoPreference_u, N_u$

   $s \rightarrow u : SSL\ Version_s, CryptoPreference_s, N_s, Sign_{CA}\{s, K_s^+\}$

   $u \rightarrow s : Sign_{CA}\{u, V_u\}, \{SSLVersion_u, r_u\}K_s^+, sign_u\{Hash(N_u, N_s, r_u)+Pad_2+$

   $Hash(Messages + u + K_{us}) + Pad_1\}$

The session key is $K_{us} = Master(N_u, N_s, r_u)$. After the key exchange, u and s negotiate a cipher (the second step):

$s \rightarrow u : \{Hash(K_{us} + Pad_2 + Hash(Messages + s + K_{us} + Pad_1))\}K_{us}$

$u \rightarrow s : \{Hash(K_{us} + Pad_2 + Hash(Messages + c + K_{us} + Pad_1))\}K_{us}$

After the secure channel is established with s via SSL, u sends a request with u's OpenID UID to web service s.

$u \rightarrow s : \{u, UID\}K_{us}$

2. s discovers the user's OpenID identity provider from the OpenID UID of user u using Yadis protocol.

   $s \rightarrow AS : ping$

   $AS \rightarrow s : ACK$

3. s redirects user u to the authentication service AS of the OpenID identity provider.

   $s \rightarrow u : \{s, AS\}K_{us}$

4. u creates another secure channel with AS.

   $u \rightarrow AS : u, SSL\ Version_u, CryptoPreference_u, N_u^{'}$

   $AS \rightarrow u : SSL\ Version_{AS}, CryptoPreference_{AS}, N_{AS}, Sign_{CA}\{AS, K_{AS}^{+}\}$

   $u \rightarrow AS : sign_{CA}\{u, V_u\}, \{SSLVersion_u, r_u^{'}\}K_s^{+}, sign_u\{Hash(N_u^{'}, N_{AS}, r_u^{'}) + Pad_2^{'} + Hash(Messages^{'} + u + K_{uAS}) + Pad_1^{'}\}$

   The session key is $K_{uAS} = Master(N_u^{'}, N_{AS}, r_u^{'})$. After the key exchange, u and AS negotiate a cipher.

   $AS \rightarrow u : \{Hash(K_{uAS} + Pad_2^{'} + Hash(Messages^{'} + AS + K_{uAS} + Pad_1^{'}))\}K_{uAS}$

   $u \rightarrow AS : \{Hash(K_{uAS} + Pad_2^{'} + Hash(Messages^{'} + u + K_{uAS} + Pad_1^{'}))\}K_{uAS}$

   u sends its OpenID and password to AS to authenticate

   $u \rightarrow AS : \{UID, K_u, s\}K_{uAS}$

5. AS creates a session id and a cookie and then redirects user u back to service s.

   $AS \rightarrow u : \{Session_{ID}, s\}K_{uAS}$

6. u sends the request to s with the session id in cookies.

   $u \rightarrow s : \{u, Session_{ID}, u\}K_{us}$

7. s establishes a secure channel with AS.

   $s \rightarrow AS : s, SSL\ Version_s, CryptoPreference_s, N_s^{'}$

   $AS \rightarrow s : SSL\ Version_{AS}, CryptoPreference_{AS}, N_{AS}^{'}, Sign_{CA}\{AS, K_{AS}^{+}\}$

   $s \rightarrow AS : sign_{CA}\{s, V_s\}, \{SSLVersion_s, r_s\}K_s^{+}, sign_u\{Hash(N_s^{'}, N_{AS}, r_s) + Pad_2^{''} +$

   $Hash(Messages^{''} + s + K_{sAS}) + Pad_1^{''}\}$

   The session key $K_{sAS} = Master(N_s^{'}, N_{AS}^{'}, r_s)$. After the key exchange, s and AS negotiate a cipher.

   $AS \rightarrow s : \{Hash(K_{sAS} + Pad_2^{''} + Hash(Messages^{''} + AS + K_{sAS} + Pad_1^{''}))\}K_{sAS}$

   $s \rightarrow AS : \{Hash(K_{sAS} + Pad_2^{''} + Hash(Messages^{''} + s + K_{sAS} + Pad_1^{''}))\}K_{sAS}$

   s then sends the session id to AS to verify

   $s \rightarrow AS : \{Session_{ID}, u, UID, s\}K_{sAS}$

8. AS confirms the session id with s.

   $AS \rightarrow s : \{ACK, Session_{ID}, UID\}K_{sAS}$

9. s decides whether or not to provide service and responses to u.

   $s \rightarrow u : \{response\}K_{us}$

The message flow of the authentication protocol is illustrated in figure 2.5.

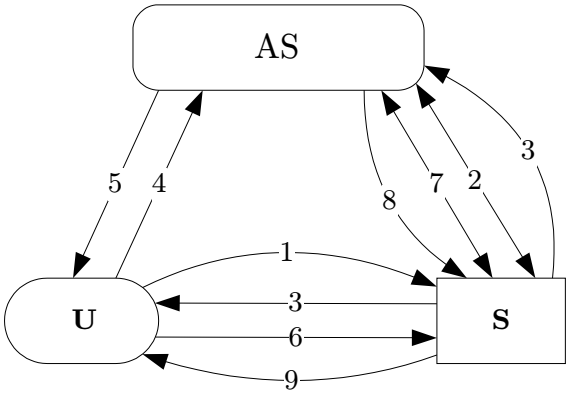


Figure 2.5: The OpenID Protocol

2.6.3 Password Authenticated Key Exchange Method (PAKE)

The PAKE method was first proposed by Steven M. Bellovin and Michael Merritt in [BM92]. The flaw in the first proposal was found and revised in [BM93]. Later, the password authenticated key exchange method and its extensions were proposed in [Jab96] [Wu98] [BPR00].

The main concept underlying PAKE was to minimise the exposure of the shared password. The authentication protocols utilise the hash value of the shared password to exchange random public keys and challenges. To prove identities, two involved parties encrypt challenges with their own private keys. The method does not require a trusted third party.

The PAKE method can be illustrated by one of its implementations, the Augmented Encrypted Key Exchange Authentication Protocol with hashed password [BM93]. Its steps are described as follows:

1. u chooses a random number r_u and sends to s:

   $u \rightarrow s : \{\alpha^{r_u} \ mod \ \beta\} h(K_{us})$.

2. s also chooses a random number r_s, uses the hash value of shared key K_{us} (h(K_{us})) to decrypt $\alpha^{r_u} \ mod \ \beta$ and computes the session key $K = ((\alpha^{r_u \ r_s}) \ mod \ \beta)$. s then generates a challenge (random number) N_s and sends to u the following message:

   $s \rightarrow u : \{\alpha^{r_s} \ mod \ \beta\} h(K_{us}), \{N_s\} K$.

3. u uses $h(K_{us})$ to decrypt the message to obtain $\alpha^{r_s} \ mod \ \beta$, calculates K, extracts the challenge N_s and responds to the challenge with the following message:

   $u \rightarrow s : \{N_u, N_s\} K$.

4. s decrypts the message to obtain N_s, and N_u and verifies N_s with the original challenge it generated in step 2. If the response is a match, s sends back the following message:

   $s \rightarrow u : \{N_s\}K$.

5. u decrypts and verifies N_s.

6. At the end of the execution, u uses its secret key and signs with the session key K.

   $u \rightarrow s : \{h(K_{us}, K)\}K$.

7. s decrypts the message to obtain $h(K_{us}, K)$ and concludes that the protocol is successful if, and only if, the signature is correct.

In the following section, these three authentication methods are evaluated by the four criteria discussed in section 2.3: security, efficiency, flexibility and scalability.

2.7 Evaluation of Existing Authentication Methods

In this section, the authentication methods discussed in section 2.6 - Kerberos, OpenID and PAKE - are assessed in terms of security, efficiency, flexibility and scalability. The assessment reveals their advantages and disadvantages in a wireless network environment.

2.7.1 The Kerberos Authentication Method

The Kerberos authentication method provides an efficient and flexible method to authenticate services. In the Kerberos authentication protocol (described in section 2.6.1), user u takes three decryptions, two encryptions and generates two nonces. Service s takes one encryption and two decryptions while the authentication service (AS+TGS)

takes two decryptions, four encryptions and generates two keys. In terms of communication costs, the protocol uses six messages for an authentication. All encryption in the original Kerberos uses symmetric cryptography. Kerberos can thus provide efficient authentication for network services. Furthermore, there are, as previously discussed, many extensions to Kerberos. The Kerberos authentication method is thus sufficiently flexible to adapt to the different requirements of wireless networks.

Kerberos does have several problems. In Kerberos and its associated extensions, there are two types of cryptography keys: session keys ($K_{u,tgs}$ and $K_{u,s}$) and long-term shared keys (K_u and K_{TGS}). While session keys are generated anew in every session, long-term shared cryptographic keys are re-usable. These long-term shared cryptographic keys are used to encrypt the messages that exchange the session keys. Although these long-term shared keys are never transferred directly between parties, the cryptography systems are vulnerable to cryptanalysis attacks.

To reduce the above risk, the common solution is to increase the key sizes of the cryptography. However, increasing the key sizes often creates higher computation and storage costs, especially in asymmetric cryptography systems. Therefore, the key sizes of long-term shared keys and the cryptography itself are major issues for Kerberos.

In addition to permanent keys, timestamping is also an issue. Most of the authentication systems in Kerberos rely on a timestamp to verify the freshness of the messages and detect replay attacks from intruders. If the clocks of clients, services and KDCs are not synchronised, the authentication systems are vulnerable under suppress-replay attacks [Gon92]. However, it is inefficient to maintain clock synchronisation for all parties involved in the authentication method regularly [BM91]. While the ticket is still valid, an adversary can re-use the ticket for obtaining unauthorised access. This feature renders Kerberos authentication vulnerable to replay attacks.

The Kerberos authentication method also experiences problems of scalability in large systems. When the number of services increases, the authentication process in

Kerberos suffers performance degradation. The process becomes less efficient because it lacks group communication support. Although Kerberos version 5 [NT94] offers multiple realms for distributed authentication to support large networks, cross-realm authentication requires explicit trust between servers and experiences high authentication costs. Due to these characteristics, the majority of Kerberos systems are seen in small network contexts [FE09].

2.7.2 OpenID

OpenID is a mechanism to provide distributed authentication for web services. It reduces the bottle-neck problem of the Kerberos authentication method by providing distributed Open ID Service Providers that verify identities for users. The method relies on SSL/TSL to establish secure communication channels between user and service, user and authentication service and authentication service and service.

To achieve security for authentication in OpenID, user u takes two asymmetric encryptions to exchange keys for SSL/TSL, thirteen symmetric encryptions and three symmetric decryptions. Service s takes one asymmetric encryption, one asymmetric decryption, eleven symmetric encryptions and thirteen symmetric decryptions. AS uses two asymmetric decryptions, ten symmetric encryptions and twelve symmetric decryptions. The protocol uses total twenty-five messages for an authentication. With a high cost in both computation and communication, OpenID is not suitable for wireless networks.

In term of security, OpenID is vulnerable under cryptanalysis attacks and phishing attacks. OpenID uses asymmetric cryptography (RSA or Diffie-Hellman in SSL) to perform key exchange. After the key exchange, u communicates with s and AS with session keys. Because both the authentication key (K_u) and other asymmetric keys used in the key exchange (K_s^+ and K_{AS}^+) are long-term shared keys, they may be vulnerable to cryptanalysis attacks. Furthermore, because they are sent to AS in the

last message of step four, the authentication keys K_u may be broken by adversaries. Le [LJCS08] and Oh [OJ08] both demonstrate successful phishing attack attempts. In these attempts, a malicious service can modify the message in the third stage of the protocol so that user u is redirected to a falsified AS. By masquerading as a valid AS, the fraudulent AS is able to obtain authentication key K_u. Thus, OpenID is vulnerable to phishing attacks.

Of the four criteria, OpenID performs best in terms of scalability. Using the distributed authentication server AS, OpenID divides users and services into different domains. Each AS handles authentication for users and services in its own domain. The OpenID method is thus able to provide authentication for large systems.

Although scalable, OpenID is not flexible. Because OpenID relies on SSL and authentication protocol, it does not allow the mechanisms required for the different platforms and resources of users and services.

2.7.3 The Password Authenticated Key Exchange Method

In terms of security, the main focus of the PAKE method is to minimise the risk of dictionary and cryptanalysis attack. The asymmetric keys (r_u and r_s) used to exchange session keys are generated randomly and are not re-usable. The hash value of the long-term key K_{us} is only used to encrypt a random public key for encryption. Because the long-term shared key K_{us} is derived from the user password, the PAKE method uses $h(K_{us})$ as the cryptographic key instead of using K_{us} to minimise the risk of dictionary attack on the user password. Despite these precautions, cryptanalysis attacks described in [WW04] [ZLR09] [CL06] [CQZ09] [PGW06] [YYC05] [YY05] are able to perform cryptanalysis on the implementation of PAKE protocols to gain unauthorised access. On the other hand, because PAKE employs mutual authentication messages, it is able to prevent phishing attacks. In [BM93], the analysis shows that PAKE is able to prevent replay attacks. However, Shim [Shi07] points out that PAKE is vulnerable to

certain protocol attacks [KSW98] (including replay attack) that derive incorrect beliefs and denial of service attacks to deceived parties.

PAKE is operationally inefficient. In its authentication process, u takes two asymmetric encryptions and three symmetric decryptions. s also takes the same number of encryptions and decryptions as u. In terms of communication cost, the authentication process requires five messages.

PAKE has limited flexibility. Some versions of PAKE use asymmetric cryptography (either Diffie Hellman [Jab96], RSA [MS99] or ECC [LW08, YS03] methods). However, while it supports asymmetric cryptography, PAKE does not support symmetric cryptography for key exchange.

PAKE has limited potential for scalability. It does not use a trusted third party and therefore does not support large systems. However, Wu and Zhu [WZ08] propose a method using trusted third parties to perform distributed authentication. The authentication extension concept is based on the crossed realm authentication in Kerberos. The cost of crossed realm authentication in this extension is high; efficiency is traded for scalability.

2.7.4 Evaluation Conclusion

In this section, the three authentication methods are compared in terms of security, efficiency, flexibility and scalability and the results of our evaluation are presented.

The findings of the previous section are summarised in table 2.4.

Table 2.4: Analysis of Existing Authentication Methods

Authentication Method	Security	Efficiency	Flexibility	Scalability
Kerberos	No	Yes	Yes	Limited
OpenID	No	No	No	Yes
PAKE	No	No	Limited	Limited

As can be seen, no single authentication method substantially meets all criteria. Kerberos has good efficiency and flexibility, but suffers limited scalability and low security. It is vulnerable to replay attacks and cryptanalysis attacks. OpenID rates well only on the criteria of scalability while PAKE suffers security vulnerability, limited flexibility and scalability and poor efficiency.

In conclusion,

- the authentication methods listed in table 2.4 are either scalable or flexible (but not both), and achieve security only at a high computational cost;
- authentication methods with high efficiency do not provide sufficient security;
- there is a trade off between security and efficiency; and
- no single existing authentication method possesses the four properties necessary for large, dynamic wireless networks and their users and services.

To fill this void, in chapter three we propose an authentication model that is secure, efficient, flexible and scalable, and is suitable for both individual and group wireless network users and services.

2.8 Summary

In this chapter, existing authentication methods and their related techniques to support authentication were introduced. Cryptography and key management are the two most basic and important techniques for authentication. Modern authentication methods employ these techniques. Of these authentication methods, three commonly used authentication methods were reviewed in this chapter. To analyse these authentication methods and the mechanisms of the proposed authentication model in the next chapter, a set of criteria comprising security, efficiency, flexibility and scalability were

used. Of these criteria, security is the most important. To assess security, common attack methods on authentication were examined. A method to validate security, SVO, was also described. The discussion showed that the three commonly used authentication methods are unsuitable to provide secure and efficient authentication for wireless network users and services.

Of the basic techniques, cryptography plays an important role in securing authentication. There are two major types of cryptography: asymmetric and symmetric cryptography. Each type has its own advantages and disadvantages. While the major weakness of symmetric cryptography is the requirement of a shared cryptographic key that relates to key exchange and key management, its strength is its low computational cost. In contrast, asymmetric cryptography does not require a pre-shared cryptographic key. However, the computational cost of asymmetric encryption and decryption is much higher than the cost of symmetric cryptography. To manage cryptographic keys used to secure group communication and authentication keys, the logical key hierarchical scheme was described. Two other cryptographic types used in authentication, the one-way hash cryptography and the one-way pad, were also reviewed.

Three common authentication methods used for wired network services - Kerberos, OpenID and PAKE - were examined. To evaluate these authentication methods, four criteria were proposed: security, efficiency, flexibility and scalability. Of these, security is the most important. The security features of the authentication methods were evaluated by their ability to cope with common attacks: replay attacks, cryptanalysis attack and phishing attacks.

The evaluation results showed that the existing authentication methods do not provide secure, efficient, flexible and scalable authentication for wireless network users and services. While Kerberos is an efficient and flexible authentication method, it is not secure and has some limitations in scalability. In contrast, OpenID was built

for large authentication systems. However, OpenID is neither efficient nor sufficiently flexible for wireless networks. It is also vulnerable to phishing and cryptanalysis attacks. Although the PAKE method has a strong cryptography system and secure authentication, it has limited efficiency and scalability. Furthermore, the PAKE method is vulnerable to cryptanalysis attacks. In conclusion, no existing authentication method supports secure, efficient, flexible and scalable authentication for wireless network users and services.

In the next chapter, we propose an authentication model designed to overcome these shortcomings and to provide authentication for individual and group wireless network users and services.

Chapter 3

AMUS Authentication Model and Architecture

We showed in the previous chapter that the current authentication approaches do not provide authentication that is both secure and efficient for services in wireless networks. Because of the nature of signals broadcast in the air, communications in wireless networks are more vulnerable to security attacks (described in section 2.4) than wired networks [Mil01] [Hen03]. However, strengthening the authentication of mobile devices in wireless networks by increasing cryptographic key sizes can result in high computation costs [BDR+96], especially in asymmetric cryptography [WGE+05] [Lop06]. Current mobile devices often have limited resources [Tar03] such as low-power processors, small memory, limited battery power and low network bandwidths. Providing authentication for wireless network users and services that is both secure and efficient is therefore challenging.

Flexibility and scalability are also of concern [LB01] [JTY97] when applying existing authentication methods to wireless networks. Because users and services in wireless networks are dynamic, the current authentication approaches may become unfeasible due to the high costs related to management, storage and search operations

on high numbers of identities in large wireless networks. Operation costs that are minor in small-scale authentication situations may be magnified to constitute a large part of the authentication cost in large-scale authentication situations. In addition, a specific authentication method that suits one situation may not suit other situations in wireless networks. Because of the limited resources of mobile devices, not all mechanisms can be used in all mobile devices. An authentication model is required that can support multiple mechanisms in order to adapt to different mobile devices and different situations.

Grouping is an approach commonly used to solve the efficiency problem in large systems. The grouping of users [CCL05] [FKC03] [CS05], applications [JLP97] [Haa08] and requests [KM02] [TLP03] is commonly used to achieve scalability in large systems. In authorisation processes, user groups have been employed in Unix [GS03] and Windows [CG06]. Role Based Authorisation Control [SCFY96] [FKC03] develops the user group concept into a "role" in order to decrease management costs and to improve efficiency for authorisation updates in large systems. Meanwhile, authorisation for group collaboration [PWFK02] [JL96] and distributed applications [Var05] [Zur96] also works with groups of adjacent network nodes to achieve security and efficiency in scalable authorisation systems.

Authentication involving groups of users and services (as opposed to individual users and services) presents further issues as no existing method offers both secure and efficient authentication for groups of users and services in large systems. Although grouping users and services is common and straightforward, providing secure, efficient and flexible authentication for them is difficult. Because current authentication methods are neither secure nor efficient for user groups and service groups, some efforts [AST00] [Han00] [BCEP04] have been made to propose group authentication methods. However, while these methods support authentication for user groups, they neglect support for service groups. Moreover, these methods are vulnerable to

replay and man-in-the-middle attacks [JP02] [PQ02] [NKW04]. Furthermore, the dynamic nature of user group and service group members in wireless networks renders the issue of balancing security and efficiency in authentication for both individuals and groups of users and services problematic.

In this chapter, a new authentication model for wireless network users and services, named **AMUS** (**A**uthentication **M**odel for wireless network **U**sers and **S**ervices), is proposed in order to overcome the shortfalls of the current authentication methods described in section 3.1. In the model, a collection of relationships among the basic elements of the authentication model is formed. Based on this collection of relationships, two main components, a group manager and an authentication controller, perform authentication verification for users and services in the model. The group manager is used to handle membership management of user groups and service groups while the authentication controller conducts the authentication verification process in the model. Based on the authentication model, an authentication architecture is also derived to serve as a guide to realise authentication.

The authentication architecture described in section 3.2 has two layers: the key management layer and the authentication layer. In the key management layer, authentication keys are distributed among group members. This layer is used to support the authentication verification process performed in the authentication layer. Although the two layers are linked, the authentication layer is transparent to the key management layer. The two layers thus enable flexibility as implementations can use different mechanisms in each layer.

Further discussion in section 3.3 shows that the AMUS model is able to provide flexible and scalable authentication for users and services in wireless networks. Of the four evaluation features (security, efficiency, flexibility and scalability), security and efficiency depend on mechanisms used to realise the AMUS authentication model. Scalability is enabled by the model's ability to adapt to rapid growth and shrinkage

in large wireless network users and services. Due to the model's structure, multiple mechanisms can be applied in the authentication model to suit different configurations and security requirements, thus enabling flexibility.

3.1 The AMUS Authentication Model for Wireless Network Users and Services

Before a formal description of the proposed authentication model is presented, an overview and description of the basic entities (including users, services and the user group and service group) in the model are given. Later, the formal description of the model includes definitions of general concepts, the authentication message and components of the authentication verification.

3.1.1 An Overview of the AMUS Authentication Model

Let the authentication model have n users and m services. These are grouped into x user groups and y service groups respectively. Notation for users, services, user groups and service groups are specified as follows:

- n users written as $u_1, \ldots, u_n$;
- m services written as $s_1, \ldots, s_m$;
- x user groups denoted as $UG_1, \ldots, UG_x$; and
- y service groups denoted as $SG_1, \ldots, SG_y$.

Grouping users and services can enhance scalability and efficiency without sacrificing security for authentication and authorisation. Assembly by groups is a natural characteristic of users and services. In authentication and authorisation, users are grouped according to their roles in organisations. No constraints are attached to

grouping services; services can be grouped as long as they share the same security requirements. Because of the freedom of grouping users and services, a user or service may belong to more than one group. While the privileges of users can be dynamic, roles are generally more stable. This stability reduces management costs and enhances scalability.

The underlying concept of the AMUS authentication model is to provide authentication for user groups and service groups. Unlike traditional authentication models representing individual users and services, AMUS connects users with user groups and services with service groups. By grouping users and services into user groups and service groups, individual users in the system become members of user groups. Individual services also become members of service groups. Authentication for users and services employs group identities instead of individual identities. Authentications are therefore considered to be performed for user groups and service groups, rather than for individuals or individual services. Hence, the individual authentication problem becomes a group authentication problem. The conceptual model of AMUS is illustrated in figure 3.1.

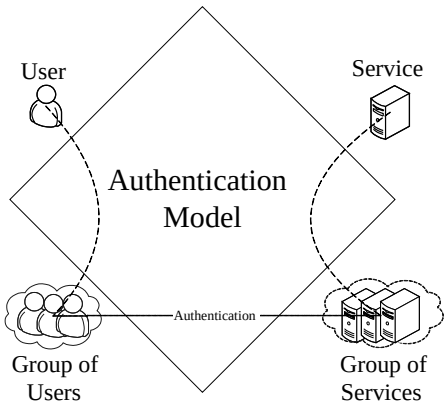


Figure 3.1: The Authentication Conceptual Model

Because the AMUS authentication model has a simple structure, it requires expansion before it can be applied to wireless networks. While group concepts have been known previously (see earlier discussion in section 1.1), users and services in wireless networks can be grouped in many different ways. Furthermore, because

group memberships of wireless network users are large and dynamic, it is essential that the AMUS model include a comprehensive authentication component to manage these memberships securely and efficiently. This core authentication component allows the AMUS model to meet its strong authentication goals.

The expanded authentication model comprises basic elements (users, services, user groups and services groups) and components (an authentication message, a collection of relationships, a group manager and an authentication controller). During authentication, users and services create authentication messages and exchange them via communication channels. An authentication message usually comprises authentication data and non-authentication related data. In order to verify the claimed identities in the authentication data of the authentication messages, we propose three components for authentication verification: the collection of relationships, a group manager and an authentication controller. The collection of relationships and the group manager are created to support different ways of grouping users and services and authentication. Based on these relationships and the group manager, the authentication controller is designed to protect users and services in the authentication model. The basic elements and components of the AMUS model are illustrated in figure 3.2.

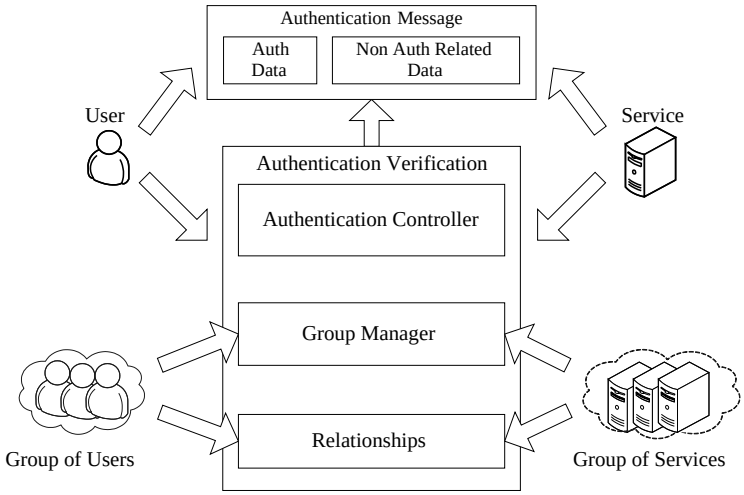


Figure 3.2: The Overview of Elements in the AMUS Authentication Model

3.1.2 General Concepts for Authentication

The basic concept of any authentication involves a user, a service, an identity and an authentication request. Usually the user is the source of an authentication request while the service is the target. In other words, the user expresses a desire to access the service and claims ownership of the identity in the authentication request. In the AMUS authentication model, each user is assumed to have a single unique user identity. Therefore, an individual user identity, denoted as U, also refers to the user owning the identity. Let n be the number of users in the system. The set of users in the system is symbolised as $\mathbb{U} = \{U_1, U_2, \ldots, U_n\}$.

The service is also a common concept in wireless networks. In [Bel08], network services (commonly shortened to "services") are defined as software modules that offer sets of utilities to provide access to resources. Among services, Web service [Wiknd] is currently one of the most developed service types. In traditional wired network systems [TvS06], services are normally tied to specific nodes to receive user requests via networks. In wireless network systems, services become more dynamic. Instead of being statically located at dedicated nodes, services are able both to roam among different wireless networks and also to migrate between different mobile devices.

Similar to users, each service has a unique service identity denoted as S. Let m be the number of services in the system. The set of users in the system is symbolised by $\mathbb{S} = \{S_1, S_2 \ldots, S_m\}$. In the model, the service identity is used to determine the service's validity and right to operate in the system. Service authentication is therefore crucial in the AMUS authentication model. Service authentication protects users from the phishing attacks of illegal services. In short, service authentication prevents malicious services from operating in systems.

In the AMUS model, users are grouped together to form *user groups*. A user group is not a new concept in security. User groups have been used in access control for operating systems [Ray03], in authorisation control [SCFY96] and in group authentication [MCR04] [DGS98]. User groups are usually derived from existing groups of users in reality such as groups of friends, employees of a company, students in a class or members of a family. Users are also grouped by their roles in a system. Usually a role correlates to a privilege in the system. In other words, users are grouped by their authorities. A user group UG of k members, $U_{i1}, \ldots, U_{ik}$, is formalised as follows:

$$UG = \{U_{i1}, \ldots, U_{ik}\}, k \in \mathbb{N}, 0 \leq k \leq n, U_{i1}, \ldots U_{ik} \in \mathbb{U} \tag{3.1}$$

Let x be the number of user groups in the system. The set of all user groups in the system is written as $\mathbb{R} = \{UG_1, \ldots, UG_x\}$.

By joining a group, users share permissions with other members in the same user group. In the model, each user group has its own privileges via a represented role. The privileges stand for permissions to access different services in the system. Because members can obtain privileges from their user groups, those who are in a user group share privileges. To ensure the privileges are given only to authorised users, each user group has specific security requirements. A user wanting to become a member of the user group has to satisfy the security requirements of the group. In this model, users in a user group can therefore use a single authentication key from the group to authenticate to services.

The service group is a new concept introduced in the AMUS model. In the model, like users, services are collated into groups. A service group, SG, of l services $S_{j1}, \ldots, S_{jl}$, is formalised as follows:

$$SG = \{S_{j1}, \ldots, S_{jl}\}, l \in \mathbb{N}, 0 \leq l \leq m, S_{j1}, \ldots S_{jl} \in \mathbb{S} \tag{3.2}$$

Services in a group share the same security requirements. Membership of a service group requires a service to grant permission to members of authorised user groups to access the service. Services are therefore grouped by security requirements. A service can belong to multiple service groups. Although these service groups may have different security requirements, the service adopts the security requirement from the service group accessing the authentication.

Services are often grouped by existing application packages [Haa08]. For example, a multimedia company provides multiple 'pay TV' channels as services. Although they are classified into categories (such as music, movie, news, fashion, cartoon or adult), the television channels are grouped by paid packages. When a user subscribes to a package, the user is able to access channels in the package. Instead of controlling authorisation and authentication for individual channels, the company manages its services via service packages.

Taking another example, an Enterprise Resource Planning (ERP) [EP01] system used to control and manage the business processes of a company includes many services for different operations in different business processes. The complexity of the business requires numerous services. The services are grouped into modules defined as software packages such as human resources, project management, supply chain management and customer relationship management. Each package has services to support business activities for the company. From a security perspective, these packages are considered service groups. To efficiently manage authentication and authorisation for these services, the security system controls service groups (packages) instead of individual services.

Security requirements of services are derived from their service groups. By accepting membership of a service group, a service agrees to share security levels with other services in the same group. A service's security may be compromised if the service is assigned into an inappropriate service group. When a service belongs to more

than one service group, the security requirements of the service depend on the active service group in a specific authentication.

In order to provide authentication for users and services, the AMUS model includes authentication verification as a core component. This component performs authentication by validating claimed identities in authentication messages.

3.1.3 The Authentication Message

An authentication message (also called an authentication request) comprises two parts: elements related to authentication and elements not related to authentication. The authentication-related elements include authentication entities and their claimed identities. This element type also includes the supporting data for authentication such as authentication keys, nonces (random numbers to validate the freshness of message) and time stamps. The elements not related to authentication comprise the parameters of the request used to perform tasks in the service. Because these elements are irrelevant to the scope of the authentication model we exclude them from the model. Only the authentication-related elements of entities, authentication entities and supporting data are further discussed.

Entities

In an authentication message, entities are both the source and the target of an authentication request. The source is the entity making the request while the target is the entity receiving the request. In most cases, the source is the user and the target is the service. However, it is possible for the source and target of an authentication request to be either or both user and/or service. In the AMUS authentication model, the term entity can refer to either user or service.

Claimed Identities

An identity is claimed from the entities involved in an authentication message. In an authentication request, the source usually claims its identity by attaching the identity in the message for authentication and authorisation. In the AMUS model, users and services do not use their individual identities for authentication. Instead, their group memberships are used as their identities for authentication. Users utilise their user groups while services utilise their service groups as their authentication identities. In other words, during an authentication, a user or service claims that it is a member of a user or service group in an authentication request. The claimed identities are also the group identities of a user or service group.

These three elements entities, claimed identities and authentication-related data together with task-related data (not discussed) form the authentication message. The authentication-related data includes nonces, timestamps, tickets, authentication keys and other supporting materials for authentication. The existence and combination of these data to support authentication verification depend upon implementation of authentication messages in the protocol. Therefore, authentication messages are subject to authentication verification.

3.1.4 Authentication Verification

The AMUS model has authentication verification as a core component. The authentication verification itself has three main parts: a collection of relationships, a group manager and an authentication controller. The tasks of the group manager and the authentication controller carry out the relationships between entities and identities. To perform authentication verification, the group manager and the authentication controller need to identify the relationships that exist between entities and identities. The collection of relationships between entities and identities is thus fundamental to the group manager and the authentication controller in the model.

The Collection of Relationships

Within an authentication domain, a number of relationships exist. The authentication relationship between a user U and its identity is called a user authentication relationship or UA. Similarly, the authentication relationship between a service S and its identity is called a service authentication or SA. Between user U and user group UG, the user group assignment relationship is denoted UGA. Correspondingly, the relationship between entity service S and identity service group SG is called a service group assignment or SGA. In an authentication request, when the source is a user and the target is a service, between two identities UG and SG, the group authentication relationship is GA. The collection of relationships between entities and identities is illustrated in figure 3.3.

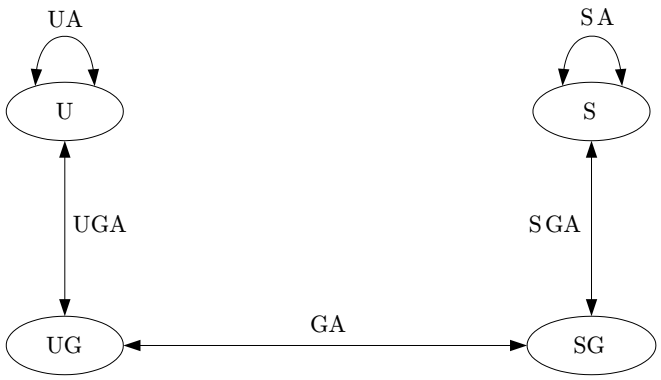


Figure 3.3: The Relationship Between Entities in the Model

Relationship 3.1. *User Authentication*

A user authentication (UA) is a relationship between a user and its individual user identity U at the group manager level. UA describes the authenticity status after the user performs an authentication with the group manager. The status explains that the group manager trusts the claimed identity U of the user. The user authentication relationship is formalised as follows:

$$UA \subseteq user \times \mathbb{U}, \text{a many-to-many, user-to-identity relationship} \tag{3.3}$$

The user authentication relationship is achieved only when the user proves its identity to the group manager in a user group join request. Ownership of a shared individual authentication key with the group manager constitutes evidence. By demonstrating the ability to encrypt data using the individual authentication key, the user creates trust in its identity and establishes a user authentication relationship.

After obtaining authenticity status, U is assigned to the user groups of which it has membership. In other words, the user group assignment UGA relationship is established only after user authentication.

Relationship 3.2. *User Group Assignment*

The relationship between a user and a user group is called a user group assignment (UGA). UGA enables a user to become a member of a user group. The UGA relationship is formalised as follows:

$$UGA \subseteq \mathbb{U} \times \mathbb{R}, \text{a many-to-many, user-to-user group assignment relationship} \quad (3.4)$$

In the AMUS model, a user can belong to multiple user groups. A user can thus have multiple roles in the system. Although user groups have varying security requirements, different authentication keys are used for different groups. This prevents security from being compromised for the different user groups. If a user has more than one membership, it has to use different keys for authentication. In the example of the software development company in figure 3.4, Robin, Harry and Michael are three users while the three user groups are based on three roles: manager, programmer and test engineer. In this example, each user is a member of more than one user group. In the example, user Harry has memberships with both the developer and the tester groups. The figure shows that Harry can be either a developer or a tester.

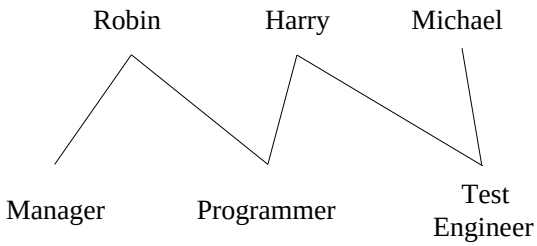


Figure 3.4: The User Group Assignment Relationship

Relationship 3.3. *Service Authentication*

Similar to user authentication, a service authentication (SA) is a relationship between a service and its service identity S at the group manager level. SA describes the authenticity status after the authentication process is performed between the service and the group manager. The status explains that the group manager trusts the service identity that S has claimed. SA warrants that the service is not fraudulent. A service authentication relationship is formed after the service authenticates its identity through a service group join request. The service authentication relationship is written as follows:

$$SA \subseteq service \times \mathbb{S}, \text{a many-to-many, service-to-its-identity relationship} \tag{3.5}$$

SA protects users from the security attacks of malicious processes masquerading as legitimate services. In the AMUS model, only legitimate services are able to respond to requests from users in the system. In phishing attacks on OpenID authentication [LJCS08], malicious services masquerading as legitimate services were able to steal private personal information (including authentication keys). Using the AMUS authentication model, when a service fails to authenticate its identity, the service authentication relationship is not formed. Consequently, under this model, fraudulent services are detectable.

Relationship 3.4. *Service Group Assignment*

A service group assignment (SGA) describes that a service S is a member of a service group SG. In other words, service S is assigned into the service group SG.

The relationship is formally defined as follows:

$$SGA \subseteq \mathbb{S} \times \mathbb{P}, \text{a many-to-many, service-to-service group assignment relationship} \tag{3.6}$$

Similar to UGA, this relationship allows a single service to have multiple memberships of service groups. Each service group has a different security level. For authentication, each service uses the group key that correspond to a particular service group. Figure 3.5 illustrates an example of a service assignment for two service groups named code and tester respectively.

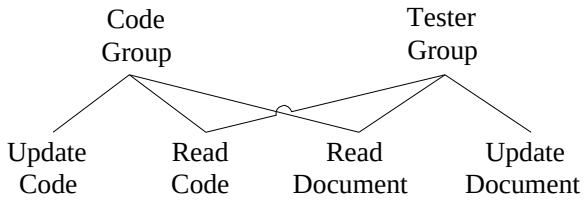


Figure 3.5: The Service Group Assignment Relationship

In this example, four services are named: update code, read code, update document and read document. These services are grouped into two service groups: the code group and the test group. The code group contains three services: update code, read code and read document. When a developer can access the code group, that person is able to update and read code. The developer is also able to read the document but is not able to update the document.

Relationship 3.5. *Group Authentication*

A group authentication (GA) is built from two types of relationships:

i. a trust relationship of the claimed user group identity of a user; and

ii. a trust relationship of the claimed service group identity of a service.

These are formally defined as follows:

$$\begin{aligned} TUR \subseteq \mathbb{U} \times \mathbb{R}, \text{ a relationship between a user and a user group; and} \\ TSR \subseteq \mathbb{S} \times \mathbb{P}, \text{ a relationship between a service and a service group.} \end{aligned} \tag{3.7}$$

By default, when a user and a service are the source and the target of the authentication respectively, a group authentication relationship GA is defined as a combination of two trusted relationships GA_1 and GA_2 where GA_1 is from TUR and GA_2 is from TSR.

$$GA = GA_1 + GA_2, GA_1 \in TUR, GA_2 \in TSR \tag{3.8}$$

The AMUS authentication model also can be extended to authenticate between a user and another user, a service and a user or a service to another service. Therefore, the extended group authentication relationship GA^* can be defined as a combination of two trusted relationships GA_1 and GA_2 where GA_1 and GA_2 are from $TUR \cup TSR$, as follows:

$$GA^* = GA_1 + GA_2, GA_1, GA_2 \in TUR \cup TSR \tag{3.9}$$

The combination of the two relationships GA_1 and GA_2 represents mutual authentication for the group authentication GA. The authentication relationship requires a mutual trust between the source and the target of the authentication. Because GA_1 is responsible for forward trust, it also named as the forward group authentication relationship. This relationship explains that the source has authenticated its claimed group identity with the target and has thus created trust. Similarly, GA_2 uses the backward group authentication relationship to explain that the source can also trust the claimed identity of the target. The group authentication relationship is established only after both relationships GA_1 and GA_2 are established. The condition warrants that the group identities claimed from both the source and the target in the request have been successfully verified. This mutual trust helps authentication prevent phishing attacks from fraudulent services.

In most of the authentication requests, the source is a user U and the target is a service S. The group authentication relationship GA is established from the forward group authentication relationship GA_1 (created from the authentication of user group identity UG of U at service S) and the backward group authentication relationship GA_2 (which represents trust of U with the claimed service group identity SG of service S).

In another example, service S_1 tries to access service S_2 by making an authentication request. In this request, the source is a service S_1 and the target is another service S_2. While SG_1 and SG_2 are the claimed identities of service S_1 and S_2 respectively, the group authentication relationship GA^* is re-written as follows:

$$GA^* = (S_1 \rightarrow SG_1) \vee (S_2 \rightarrow SG_2) \tag{3.10}$$

This expression explains that both services S_1 and S_2 have to authenticate their claimed identities before the mutual group authentication relationship can be formed.

Based on the collection of relationships, other components in the authentication verification process can be built. The group manager is built from the UA, UGA, SA and SGA relationships. The authentication controller is built from the GA relationship. Figure 3.6 illustrates the entities and components, and the relationships between entities and components, in the model.

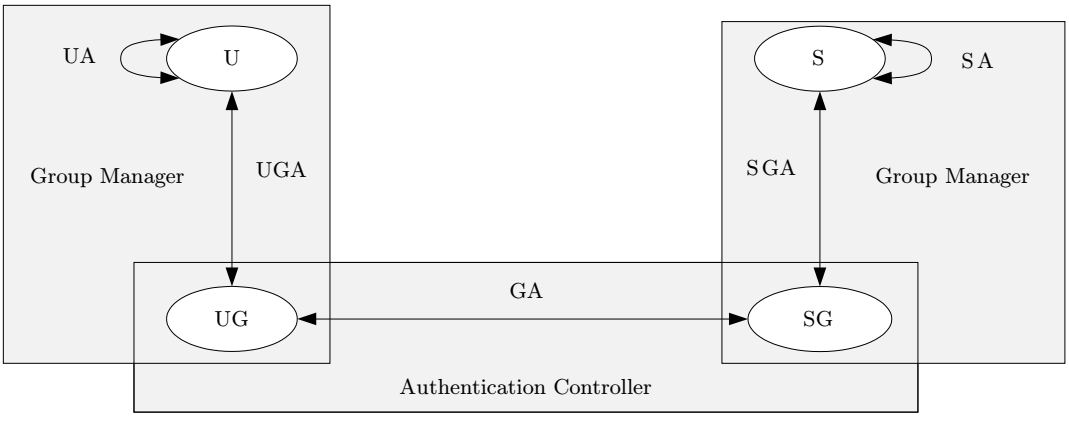


Figure 3.6: Summary of Relationships between Entities, Elements and Components

The Group Manager

The group manager is built from two relationships: the user group assignment (UGA) and the service group assignment (SGA). These relationships involve the two entities of a user and a service. The group manager is therefore divided into two sub-group manager components: the user group manager and the service group manager. The user group manager supervises the UGA relationship while the service group manager handles the SGA relationship.

Component 3.1. *User Group Manager*

The user group manager (UGM) is a key management service designed to manage memberships of user groups. This service handles group operations including group join and group leave requests from users in the system. At the initial stage, before authenticating with services in the model, users have to join appropriate user groups. At this stage, the UGM verifies the individual identity and memberships of users. The status after successfully verifying a user identity is named user authentication. This status forms the user authentication relationship (UA). In other words, the UGM controls the UGA relationship and the UA relationship.

Component 3.2. *Service Group Manager*

The service group manager (SGM) is another key management service used to manage membership of service groups. Similar to the UGM, the SGM handles group operations from services in the model. The service authentication relationship forms the SGA relationship. In summary, the SGM manages two relationships in the AMUS model: the service group assignment SGA and the service authentication SA.

The Authentication Controller

The authentication controller is the core component that verifies authentication requests for wireless network users and services in the AMUS model. The centre of

the authentication controller is an authentication verifier. To verify authentication requests, the authentication verifier employs authentication protocols. Because services and users usually have different characteristics and security requirements, using one authentication protocol for different users and services is not appropriate. In AMUS, the authentication verifier chooses an authentication protocol that suits the characteristics and requirements of both services and users. The authentication controller employs a special service, called an authentication service, to handle the authentication process.

Component 3.3. *The Authentication Service (AS) is a specific service used to verify the claimed identities of entities in requests.*

The authentication service AS is a trusted third party that verifies authentication requests from users and services in the model. Unlike previous authentication approaches where authentication is tightly integrated with services, in the AMUS authentication model the authentication module is separated into a service, similar to the Key Distribution Centre in the Kerberos authentication model. It is assumed that users and services in the AMUS authentication model have the full trust of AS. Trust is represented by sharing authentication keys between users and AS or between other services and AS.

Each authentication entity (either user or service) shares a secret with AS. The shared secret is only known by AS and the entities. Shared secrets are usually implemented as authentication keys. When an entity wants to prove its claimed identity, it has to show evidence that it knows the shared secret (the authentication key). AS verifies the claimed identity of the entity by creating a random number, called a nonce, and challenging the entity to encrypt this nonce. Apart from AS, only valid entities have the authentication key. Hence, if the nonce is correctly encrypted by the authentication key, the entity can create trust in its claimed identity for AS. The verification process is called authentication.

In the AMUS model, authentication identities verified by AS are group identities instead of individual identities. These claimed identities are either user group UG or server group SG. The authentication of a user group identity from a user identifies whether a user is a legitimate member of user group UG. The evidence of user group identities and service group identities is represented by shared group keys obtained from the group manager. These group keys are used as authentication keys for user groups and service groups.

To validate the group identity from the authentication keys, authentication protocols are used by the authentication service.

Authentication Protocols

Definition 3.1. Authentication protocols are defined in [BFMT03] as cryptography protocols that enable an entity to prove its identity to other entities. These protocols contain a sequence of authentication messages that are sent in a specific order among the parties in the authentication. In the AMUS authentication model, authentication messages are exchanged between users, services and the authentication service. In the authentication messages, users and services try to prove their identities by showing their knowledge of authentication group keys as evidence for authentication.

An authentication protocol is formalised as follows:

$$AuthenticationProtocol = (Message_1, \ldots, Message_k), k \in \mathbb{N}, k \geqslant 1 \tag{3.11}$$

All of the above entities, identities, components and relationships are used to construct the AMUS authentication model. To manage the relationships and connect all entities, identities and components together in the model, the group manager and the authentication controller work together. Authentication verification in the AMUS model relies on these two main components (illustrated in figure 3.7). Based on these

two components, in the next section we propose an authentication architecture. The authentication architecture is used as a framework to build different authentication realisations to adapt to different security requirements of wireless network service systems.

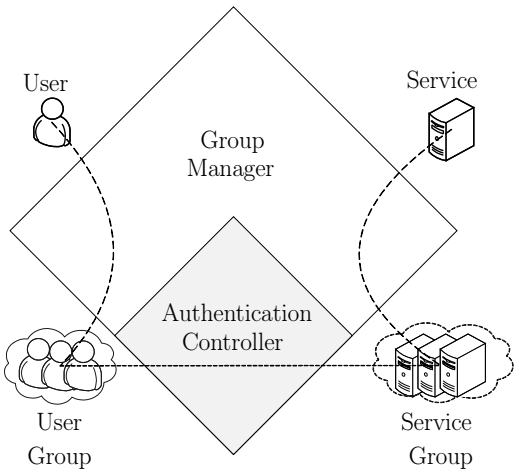


Figure 3.7: The Authentication Conceptual Model in Greater Detail

3.2 The Authentication Architecture

The authentication architecture is an abstract description to guide the realisation of the AMUS authentication model. The authentication architecture is divided into two layers: key management and authentication. These two layers are derived from the group manager and the authentication controller in the AMUS authentication model. The group manager creates a transparent layer to provide operations related to group memberships and group authentication keys for users and services. The authentication controller builds an authentication layer to control group authentication operations. This layer supports the authentication protocols in order to perform authentication. Figure 3.8 illustrates the two-layer authentication architecture.

In the architecture, each user U, a service S or the authentication service AS is integrated with a key management module and an authentication module. The key

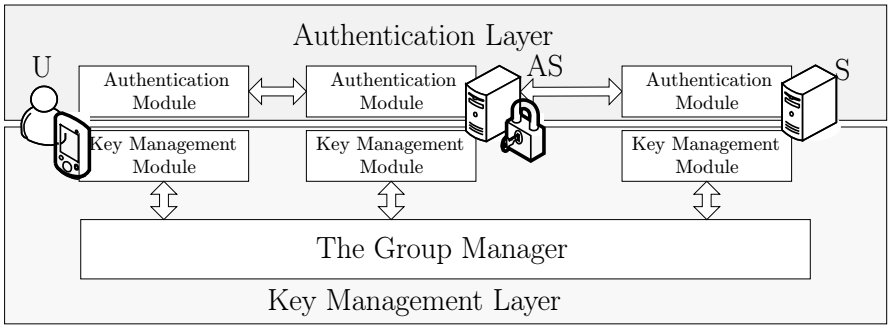


Figure 3.8: The Authentication Architecture

management module updates group keys from the group manager via the key management layer. The authentication modules use the group key obtained from the key management module to authenticate with authentication services via the authentication layer.

3.2.1 The Key Management Layer and the Group Manager

Operations of the key management layer rely on the key modules of the involved entities of the authentication model. The main function of the key management modules is to connect users to user groups and services to service groups. In order to connect to user groups, the key management module of a user associates with the user group manager UGM. Likewise, the key management module of a service associates with the service group manager SGM to connect a service to its service groups. After connecting to one group, the user or service obtains a cryptographic key for group communication related to the group. This key (the group key) is also used as a group token to validate membership to relevant user or server groups in the authentication layer. In other words, the group manager provides a facility to connect users and services to their group identities.

To connect users and services to user groups and service groups, the group manager provides two basic operations: user or service member join and leave. During the user or service member join and member leave operation, group keys are securely updated and distributed to the key manager module of affected users or services. The

group key updates in the member leave operation prevent evicted member users or services from obtaining current group keys to perform unauthorised authentication. Similarly, the group key update in the member join operation protects past communication data from unauthorised access by newly joined members.

In addition to the two basic group operations, the group manager also supports frequent handoffs of members in the wireless networks and periodical rekeying. A single handoff from a user or a service invokes one member leave operation and one member join operation. Frequent handoffs from wireless networks users and services can create overheads for key updating. We propose the group manager in order to reduce the overhead cost associated with handoff operations. Meanwhile, periodical rekeying protects group keys from cryptanalysis attacks.

After users and services are connected with user groups and service groups, their key management modules obtain group keys from the group manager. In turn, their authentication layer uses the group keys as authentication keys to prove their group identity. In summary, the group manager operates as a transparent key management layer to update, synchronise and secure authentication keys for members of these groups. The operations of this layer are independent from the authentication controller. The layer focuses primarily on providing security for key distribution.

3.2.2 The Authentication Layer and the Authentication Controller

The authentication controller performs authentication in the authentication layer via the authentication modules of involved entities. There are three types of entities involved in the authentication layer: user U, service S and authentication service AS. Among these, the authentication service AS must be trusted by all users and services; it is a trusted third party to verify authentication requests from users and services. The trust indicates that the authentication keys of user groups UG and SG are known by the authentication service AS. In contrast, users and services do not share their

authentication keys. Consequently, only the authentication service is able to perform authentication for members of both user groups and service groups.

3.2.3 The Authentication Protocol

The authentication protocol plays a crucial role in verifying authentication in the authentication layer. Authenticity status [WL92] is only achieved once the current authentication protocol run verifies that the response messages are within the validity windows. The verification process normally involves checking nonces (random numbers used as authentication challenges), timestamps and a ticket or token. The goal of the authentication protocol is to create trust in the identity of one (or more) communicating entities. The trust manifests as a secure channel that is established through the protocol run. The protocol and the checking process are conducted by the authentication modules of the involved parties (that is, the users, the services and the authentication services).

An authentication protocol based on Kerberos is proposed to complete the authentication architecture. Many possible authentication protocols could be employed and realised in the AMUS authentication model. Kerberos, OpenID and PAKE, for example, could all be employed. Of these, the Kerberos protocol is the most efficient and is also an open standard. For the architecture, we apply a simplified version of Kerberos using authentication tickets to perform authentication for a group of users and a group of services.

The authentication protocol has five steps. In the first step, user U sends a ticket request to AS to authenticate with S and claim that it is a member of user group UG. In the second step, AS responds to the request from U with a ticket ($\{UG, S, T_{AS}, \}K_{SG}$) and the session key $K_{session}$ encrypted by K_{UG}. In the third step, U uses the ticket received from AS to authenticate with S and sends a challenge (N_U) to S. In the fourth step, S responds to the challenge and encrypts the response with the session

key $K_{session}$ to confirm its identity. During this step, S also sends another challenge (N_S) to U to verify the identity of U. In the final step, U completes the authentication by responding to the challenge from S and encrypts the response with the session key $K_{session}$. The five steps of the protocol are formalised as follows:

1. $U \rightarrow AS : UG, s.$
2. $AS \rightarrow U : \{K_{session}\}K_{UG}, \{Ticket\}K_{SG}.$
3. $U \rightarrow S : \{Ticket\}K_{SG}, Challenge_u.$
4. $S \rightarrow U : \{Response_s\}K_{session}, Challenge_s.$
5. $U \rightarrow S : \{Response_u\}K_{session}.$

Kerberos is not the only approach that could be used in the AMUS model. We emphasise that the authentication model could, for example, also adapt the authentication methods of OpenID or PAKE. The protocol is simplified so that it can be realised in different ways when being used by different mechanisms. The proposed simplified authentication protocol in this architecture is an example, and not a constraint that limits authentication realisation from the authentication model. With its simplified authentication protocol, the authentication model offers flexibility in wireless networks.

3.2.4 Summary

In summary, the framework of the authentication architecture divides the model into two separate layers. The authentication layer contains the authentication controller while the key management layer contains the group manager. The authentication controller and the group manager both rely on the collection of relationships to manage group memberships and provide authentication. Using the authentication protocols, the authentication controller can examine authentication messages and verify the claimed identities of involved entities. The summarised authentication components are illustrated in figure 3.9.

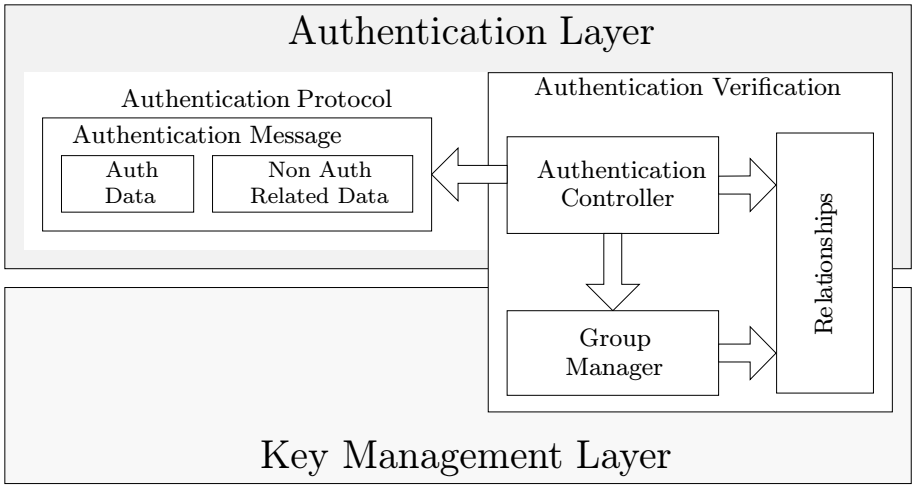


Figure 3.9: The Components of the AMUS Authentication Model

3.3 Discussion

In this section, we assess the authentication model for security, efficiency, flexibility and scalability. We discuss these criteria in order of importance. Security and efficiency are the most important properties of the authentication model. Security refers to the ability to resist attacks to protect users and services from unauthorised access. Efficiency refers to the frugal usage of computational and communication resources and storage memory. Flexibility describes the range of variation in different techniques for the authentication model and the ability to adapt to system changes. Scalability is a property that enables the authentication model to adapt to large wireless network systems and empowers it to handle changing numbers of users, services and authentication requests. These properties of the authentication model are assessed to confirm that the authentication model is suitable for wireless network users and services.

3.3.1 Security and Efficiency

The security of the authentication model depends on the security of the individual components in the model. The AMUS authentication model comprises two main processes: the authentication key distribution process handled by the group manager and the authentication verification process controlled by the authentication controller. The security of these processes therefore has implications for the security of the authentication model as a whole. A security compromise in either process could lead to unauthorised access of a system. The security of the authentication model is thus dependent upon every mechanism used to realise the group manager and the authentication controller.

In contrast to its security, the performance of the AMUS authentication model is mostly derived from the authentication verification process. With the group manager separated from the authentication controller in the model, the key management layer becomes transparent to the authentication layer. The performance of the authentication verification process is therefore independent of the key distribution process. Although authentication key distribution contributes to the total cost of the authentication model, authentication requests account for most of the computation and communication costs of the authentication verification process. Hence, the performance of the authentication model mainly depends on the performance of the mechanism used to realise the authentication controller.

It is difficult to achieve both security and efficiency using existing mechanisms to realise the authentication model. The existing mechanisms to realise the group manager or the authentication controller each have their own advantages and disadvantages in security and efficiency. For example, a Diffie-Hellman key agreement protocol consumes more computational power but is more secure than group key management. Conversely, authentication protocols using asymmetric cryptography are more secure but less efficient than those using symmetric cryptography (see earlier

discussion in section 2.1.3). Other protocols have their own strengths and weaknesses. Consequently, no combination of existing mechanisms can provide both efficient and secure authentication for wireless network users and services.

To overcome the limitations associated with existing mechanisms, in the next chapter we propose a realisation that demonstrates that the AMUS authentication model can provide authentication with both strong security and high efficiency. The realisation is built based on a dynamic key scheme and a group key management scheme in a wireless network. Both of these mechanisms rely on symmetric cryptography to offer efficient and secure protection against security attacks on wireless networks. In the realisation for the group manager, the group key management is customised so it can support authentication for user groups and service groups. Two authentication protocols using the dynamic key scheme are proposed to realise the authentication controller. Each of these is suitable for one type of wireless network services. The dynamic key scheme is used to strengthen the security of the authentication protocols to resist authentication attacks. An analysis and discussion are undertaken in chapter 5 to verify the adaptation of the AMUS authentication model in wireless networks by examining the security and efficiency of the proposed realisation presented in chapter 4.

3.3.2 Flexibility

The two-layer architecture of the authentication model enables a range of realisations. Because of the separation and transparency between the two layers, each of the layers can be independently realised and implemented. A realisation for the authentication controller in the authentication layer does not restrain the realisation for the group manager in the key group layer and vice versa. Both the group manager and the authentication controller have many possible realisations and implementations. The

number of possible combinations from the different realisations of these two components is therefore large.

Several methods are available to effectuate the group manager in the key management layer. Available candidates are the Perfect Secret Sharing Scheme [DGS98] [Dij95], the Key Distribution Scheme [BSH+98] and group key management. Of these, group key management is the most effective method to secure user and service groups. Group key management provides a means to update and distribute group keys among group members. It also offers support for rekeying operations brought about by support member changes. Moreover, besides distributing and updating keys among group members in the key management layer, group key management can also be used to secure group communications.

Both symmetric cryptographic authentication protocols and symmetric authentication protocols can be used to effectuate the authentication controller. In the authentication layer, symmetric cryptographic authentication protocol such as the Kerberos protocol [NT94], one-time password authentication [Hal94] [LB07] or the RADIUS protocol [Luo08] can provide efficient authentication schemes for groups of users and services. Likewise, asymmetric cryptographic authentication protocols such as the Needham Shroeder public key protocol [DS81], the OpenID protocol [Found], the OAuth protocol [Wor09] and the Hybrid Authentication protocol [CJ03] offer strong authentication in the authentication layer. However, when applied to authentication for wireless network users and services, both symmetric cryptographic authentication protocols and symmetric authentication protocols have disadvantages (described in chapter 2). To overcome these limitations, in the next chapter, a set of two protocols is proposed in a realisation for the AMUS authentication model that provides both strong security and high efficiency.

The authentication model can be applied to both wireless and wired networks. The range of available realisations for the components in both layers offers different

levels of security and efficiency. Realisations exist that are suitable, because of their efficiency, for large wireless network users and services operating on limited resource devices. Strong, secure realisations also exist that can be used to protect sensitive services and users in wired networks.

The AMUS authentication model can be applied to both group authentication and individual authentication. Existing user and service groups can use the authentication model to obtain access in wireless networks via their group identities. When a group has only one entity (either a user or a service), the authentication between the user group and the service group presents as individual authentication. In chapter 5, we present an extension of the authentication model that simultaneously supports individual identity authentication as well as group identity authentication.

3.3.3 Scalability

Two types of scalability need to be considered: structural stability and load stability. The following discussion describes how the AMUS authentication model achieves each of these.

The AMUS authentication model achieves *structural* scalability by grouping users and services. In the authentication layer, the number of user groups and service groups in a large system is typically smaller than the number of users and services. An authentication service consequently uses less space to maintain these group identities. A smaller search space also enhances the performance of the identity look-up process during authentication. Furthermore, these user and service groups are assumed to be more stable than individual users and services in a wireless network. With user and service groups playing a larger role, the ramifications of the actions resulting from the remaining individual users and services is minimised. Hence, management costs

(including storage, updating and searching costs) are reduced. Because of its low management costs and subsequent efficiency, the AMUS authentication model can achieve structural scalability.

The authentication model achieves *load* scalability from its two-layer authentication architecture. In the key management layer, load scalability is achieved by applying secure distributed approaches for scalable authentication key distribution. In the authentication layer, load scalability can be achieved via duplicated authentication services for large wireless networks. In the key management layer, using distributed approaches, user groups and service groups can be divided into multiple sub-groups based on their physical wireless networks. Each sub-group can be managed by a distributed key management service in a cell of the wireless network. Users and services in large wireless networks are thus separated into sub-groups. The work load from membership management tasks is distributed and localised in distributed key management services. When the load is increased, one or more key management service(s) can be added to share the load and thus enhance the performance of group membership management in the key management layer.

In the authentication layer, load scalability can be achieved via duplicated authentication services for large wireless networks. Instead of relying on a single authentication service, Gong presented in [Gon93] a model that used distributed and duplicated authentication services as a solution to the scalability problem in authentication. However, he also identified drawbacks of this approach: low security and high updating costs required to synchronise the database. To overcome these limitations, the AMUS authentication model, rather than depends on a distributed or duplicated database, utilises the group manager in the key management layer for secure authentication key distribution among authentication services. Using the two-layer authentication architecture, the duplicated authentication services can obtain authentication

keys in the key management layer from the group manager. In this way, the authentication keys are securely synchronised with the duplicated authentication services at a minimum cost without requiring a major modification in architecture. When the authentication load is increased, the authentication performance can be improved by either adding more computation resources for existing authentication service(s) or by adding more authentication service(s).

3.4 Summary

In this chapter, a formal authentication model for wireless network users and services named AMUS was described. The main idea underlying the AMUS authentication model is to provide authentication for users and services through user groups and service groups. While current approaches authenticate individual user and service identities, the AMUS authentication model operates on user group identities and service group identities. The model is built based on four basic elements: the user, the service, the user group and the service group.

To form a background to build up components for the AMUS authentication model, a collection of five relationships was proposed. These are the user authentication relationship, the service authentication relationship, the user group assignment relationship, the service group assignment relationship and the group authentication relationship. The user authentication relationship and service authentication relationship are created by verifying the authenticity of individual users and services at the group manager level. A relationship that maps between a user and a user group is called a service group assignment relationship. Similarly, a service is put into a service group via a service group assignment relationship. Based on these relationships, the group authentication relationship connecting an entity group and another entity group describes the mutual trust of the two entities from both the source and the target of an authentication. Based on these relationships, two further components of the

AMUS authentication model were proposed: the group manager and the authentication controller. The group manager handles the user groups and service groups in the AMUS model. Its operations are related to four relationships: the user authentication relationship, the service authentication relationship, the user group assignment relationship and the service group assignment relationship. The group authentication relationship is created by the authentication task of the authentication controller. The major task of the group manager is to distribute and synchronise authentication keys to members of user groups and service groups. Based on the authentication keys, the authentication controller performs authentication for members of user groups and service groups.

Drawn from the two components in the AMUS authentication model, an authentication architecture was proposed as a guide to help implement the AMUS authentication model. The authentication architecture has two layers: the key management layer and the authentication layer. Every entity in the model also has two modules: the key management module and the authentication module. These correlate to the two layers of the architecture. In the key management layer, the key management module performs operations and communications related to the group manager. Similarly, in the authentication layer, the authentication module uses the keys obtained from the key management layer to authenticate according to the authentication controller.

The AMUS model and its architecture achieve flexibility and scalability to provide authentication for wireless network users and services. The two-layered architecture offers flexibility for realising and implementing the authentication model. Different mechanisms can be independently used to implement components in the two layers to suit different security and efficiency requirements of systems. The scalability of the AMUS authentication model is derived via user groups and service groups. By grouping users and services, the authentication cost of the authentication service can be reduced and the authentication service can consequently adapt to large systems.

The same flexibility enables the authentication service to adapt to shrinking numbers of users and services.

The AMUS authentication model also offers security and efficiency. The extent to which these two properties exist in an authentication depends on the protocols used in the model's realisation. If existing protocols are employed, their limitations are such that no combination of existing mechanisms can provide both efficient and secure authentication for wireless network users and services. New protocols are required to overcome the limitations associated with existing mechanisms. These new protocols, presented in chapter 4, involve a group key management scheme and a dynamic key cryptography scheme.

Chapter 4

An Authentication Realisation of the AMUS Authentication Model

In this chapter, an authentication is described in order to show how to realise the authentication model proposed in the previous chapter to achieve both security and efficiency in authentication for wireless network users and services. It has previously been argued [AGG+09] [ZCY+08] [AG03] [AR07] [BWZ06] that operations in wireless networks usually experience a trade-off between security and efficiency. In other words, to achieve higher levels of security, authentication usually needs to incur higher computation and/or communication costs. To help overcome this problem, the authentication employs a dynamic key cryptography mechanism [NWL+10] and a group key management scheme [NWL+09]. The group manager in the key management layer is realised using the group key management scheme while the authentication protocols of the authentication controller in the authentication layer are substantiated with dynamic key cryptography.

The group key management scheme and the dynamic key cryptographic mechanism are the means by which the AMUS authentication model enables efficiency and security in wireless networks. However, while the proposed authentication achieves

high levels of security without sacrificing performance, it is not the only possible realisation of the AMUS authentication model; nor are group key management and dynamic key cryptography compulsory for use in other realisations.

Dynamic key cryptography and group key management have been chosen to realise the authentication model because of their balancing of security and efficiency. Dynamic keys [NWL+10] are one-time symmetric cryptographic keys. The idea of dynamic key cryptography is similar to one-time pads. However, instead of having pad/key exchange before encryptions, dynamic key cryptography has an off-line key generation scheme to generate sequences of dynamic keys for involved parties. This process helps prevent replay attacks and cryptanalysis attacks. Authentication realisation using dynamic keys can achieve security without degrading performance. Meanwhile, although group key management [CS05] is not the most efficient approach for managing and distributing authentication keys, it is an efficient and secure method to secure group communications among group members. Services using multi-casting to communicate with user groups in wireless networks can utilise group key management to secure their communications.

The structure of this chapter is organised as follows. Two main mechanisms used to realise the AMUS authentication model - dynamic key cryptography and the multiple membership group key management scheme - are reviewed in sections 4.1 and 4.2. The following sections, 4.3 and 4.4, describe how to use these mechanisms to realise group key management and the authentication controller. The chapter concludes with section 4.5.

4.1 Dynamic Key Cryptography

Cryptanalysis, mentioned in section 2.4.2, is a serious threat to authentication methods. By capturing and analysing ciphertexts from authentication, attackers may be able to break the cryptography by cryptanalysis. Because communications in wireless

networks are more vulnerable than those of wired networks, authentication in wireless networks becomes more vulnerable under cryptanalysis. Cryptanalysis attack affects both symmetric cryptography and asymmetric cryptograph. The more a cryptographic key is used to encrypt messages in authentication, the higher the risk that the key will be compromised. Furthermore, replay attacks on authentication re-use captured ciphertexts and plaintexts from previous authentication messages. Because the cryptographic key does not change, attackers may be able to replay previously captured messages to obtain unauthorised access. Session keys are not the solution to preventing cryptanalysis attacks and replay attacks. The longer the duration of a session, the more vulnerable its session key. By capturing communication messages, an adversary may be able to detect patterns in the encrypted messages to crack the ciphers. The compromise of one session key exposes all communication data in the session. In addition, key exchange protocols used to distribute session keys also rely on long-term cryptographic keys (either symmetric [NT94] or asymmetric [MSS98] [DH76]). These long-term cryptographic keys are normally authentication keys of users and services. Although the number of exposed ciphertexts from these long-term cryptographic keys is smaller, the opportunity to break the key exchange cryptography still exists [Kir07].

In the past, the major solution for enhancing security and reducing the risk of such cryptanalysis attacks was to increase the key size used in the cryptographic systems. However, increasing the cryptographic key size is not always the best solution. No matter how large the key, its cryptography is still ultimately breakable. Every cryptographic key is only secure for a certain amount of time. In addition, larger key sizes often require higher computational resources, especially in asymmetric cryptography. In practice, excessively large key sizes may admit denial of service possibilities whereby adversaries can cause excessive cryptographic processing. Large key sizes are also unsuitable for mobile devices with slow processing units and/or limited battery power.

To overcome the risk of cryptanalysis attack, we propose a use of dynamic key cryptography. The main idea of dynamic key cryptography is based on the one-time pad. Each message is encrypted by a different pseudo-random key called a dynamic key. This dynamic key is a one-time cryptographic key that is not reusable. Instead of having key exchange between involved parties at the beginning of every session, these dynamic keys are generated secretly for involved parties only once in the initial stage. Apart from the initial stage, dynamic key cryptography does not require any other key exchange. By inheriting features of the one-time pad, dynamic key cryptography can minimise cryptanalysis attack risks, resist replay attacks and secure communication.

In the next section, we explain the terminology and notation used to describe dynamic key cryptography and the generation scheme. Following the terms and notation, a dynamic key definition is reviewed and dynamic key theory is discussed. The final section introduces a family of dynamic key generation schemes including a function for first-time dynamic key generation and a revised version for repeated dynamic key generation.

4.1.1 Terms and Notation

Table 4.1 lists the notation used to describe dynamic keys and the generation scheme. This notation are not related to the notation in the above authentication model. Instead, they explain dynamic key sequences, intermediate keys and functions used in the generation scheme.

4.1.2 The Dynamic Key Concept

Definition 4.1. Dynamic keys are one-time symmetric cryptographic keys forming a sequence of keys.

Table 4.1: Dynamic Key Notation

$\{DK_i\}$	A sequence of dynamic keys
n	The number of dynamic keys in a sequence
m	The number of temporary keys used in the generation scheme
$DK_i, 1 \leq i \leq n$	Dynamic keys
s	The bit length of the dynamic key DK_i
$DK_c, 1 \leq c \leq n$	The dynamic keys in current use in the sequence
EK	A one-time encrypted cryptographic key
IK	The initial key. This key is another one-time initial key to generate seed key SK
$TK_1, \ldots, TK_m$	Temporary keys. These are used as parameters to calculate dynamic keys in the sequence at the beginning. During the process to create other dynamic keys in the sequence, these parameters are replaced one after another by previous dynamic keys.
SK	A seed key generated from IK and $TK_1, \ldots, TK_m$
$f(.)$	A polynomial function to generate dynamic keys
$f^{-1}(.)$	An inverted function of f, supposed to be a non-polynomial function
Pr	A probability function
Al	A polynomial algorithm to guess dynamic key
$p(.)$	A polynomial algorithm
NK_1, NK_2	One-time keys computed from IK, EK, DK_{n+1}, and DK_{n+2}
$h(X)$	A one-way hash function h of message X
$\oplus$	A bit-wise exclusive-OR operation

Dynamic keys in a cryptography form a sequence of dynamic keys. Each dynamic key in the sequence is used to encrypt one message. After the dynamic key is used, it is discarded and the next key in the sequence is used to encrypt the next message. Mathematically, a sequence of dynamic keys is presented as follows:

$$n \in N, n > 1, \{DK_i\} = \{DK_1, DK_2, \ldots DK_n\} \tag{4.1}$$

In theory, the number of dynamic keys in a sequence can be infinite. However, in practice, a sequence usually has a limited number of dynamic keys. To guarantee the security of the dynamic key cryptography, a sequence of dynamic keys should have the two following properties.

Property 4.1. *The sequence of dynamic keys for relevant parties must be identical.*

In a symmetric cryptographic system, the involved parties must share the same dynamic keys to encrypt and decrypt messages. The cryptographic key used by senders (entities encrypting the message) must be identical to the cryptographic key used by receivers (entities decrypting the message) so that encrypted messages can be decrypted by receivers. If the dynamic key used to encrypt a message is different from the dynamic key used to decrypt a message, receivers will not be able to access messages. The dynamic keys of a sequence thus must be identical for all involved parties.

Property 4.2. *Discarded dynamic keys that have been compromised must not create vulnerability for current and future dynamic keys in a sequence.*

The property can be explained as follows. For every polynomial algorithm Al, it is infeasible for Al to guess correctly the current dynamic key DK_c from the discarded dynamic keys $DK_1, \ldots, DK_{c-1}$. In other words, with s being the bit length of a dynamic key, the highest probability of Al guessing correctly the current dynamic key is equivalent to that of brute force attacks on DK_c. Assuming that the probability of guessing DK_c correctly using brute force is $\frac{1}{2^s}$, the property is formally re-written as follows:

$$\forall i, c \in N, 1 \leq i < c, Pr(Al(\{DK_i\}) = DK_c) \leq \frac{1}{2^s} \tag{4.2}$$

4.1.3 A Family of Dynamic Key Generation Schemes

To generate secure sequences of dynamic keys that possess both the above properties, we propose a family of dynamic key generation schemes. The proposed family of dynamic key generation schemes has two parts: the initial phase and the repeat phase.

Dynamic Key Generation Initialisation

Dynamic key generation can be divided into the four following steps (see figure 4.1):

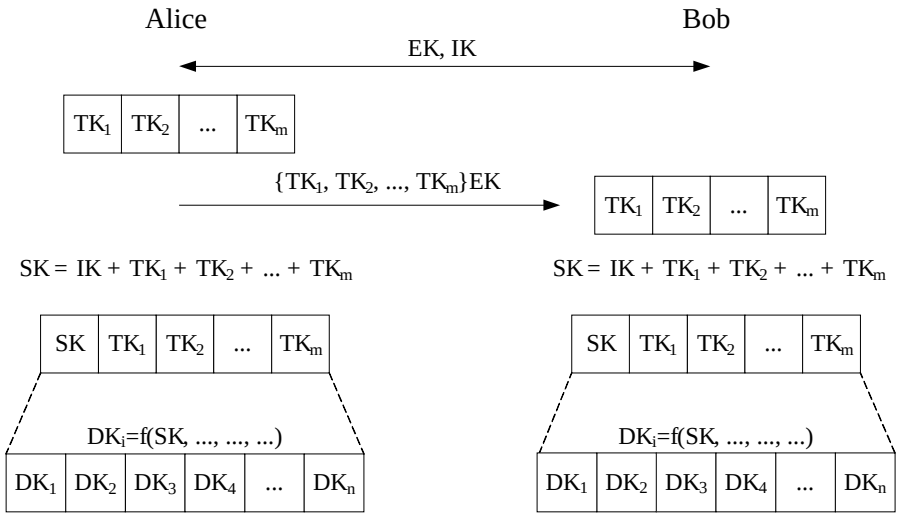


Figure 4.1: Initial Dynamic Key Generation Scheme

Step 1: Alice and Bob exchange two keys EK and IK via a secure channel.

Step 2: Alice randomly generates m initial temporary keys $TK_1, \ldots, TK_m$ and sends the message to Bob, encrypted by EK.

$$A \rightarrow B : \{TK_1, \ldots, TK_m\}EK, h(TK_1 \oplus \ldots \oplus TK_m \oplus EK) \tag{4.3}$$

The result of the hash function $h(TK_1 \oplus \ldots \oplus TK_m \oplus EK)$ is the message authentication code (MAC) used to authenticate the source of the message. The MAC verifies Alice as the sender of the message.

Step 3: Both Alice and Bob compute a seed key SK from the initial key IK and the temporary keys $TK_1, \ldots, TK_m$ using a normal addition operation.

$$SK = IK + TK_1 + TK_2 + \ldots + TK_m \tag{4.4}$$

Step 4: Both Alice and Bob generate dynamic keys.

The first dynamic key DK_1 is generated from the seed key SK and the temporary keys $TK_1, \ldots, TK_m$ by using the function $f(.)$ taking $m+1$ parameters as follows:

$$DK_1 = f(SK \oplus TK_1 \oplus \ldots \oplus TK_{m-2} \oplus TK_{m-1} \oplus TK_m) \tag{4.5}$$

With n the length of the dynamic key sequence ($n > m$), the other dynamic keys in the sequence are also generated by the function $f(.)$ with the parameters being replaced one after the other by the discarded dynamic keys as follows:

$$\begin{aligned} DK_2 &= f(SK \oplus TK_2 \oplus \ldots \oplus TK_{m-1} \oplus TK_m \oplus DK_1) \\ DK_3 &= f(SK \oplus TK_3 \oplus \ldots \oplus TK_m \oplus DK_1 \oplus DK_2) \\ &\ldots \\ DK_n &= f(SK \oplus DK_{n-m} \oplus \ldots \oplus DK_{n-3} \oplus DK_{n-2} \oplus DK_{n-1}) \end{aligned} \tag{4.6}$$

Of the available functions to implement function $f(.)$, the one-way function [KL07] is the most suitable function to implement $f(.)$ in order to compute secure dynamic keys.

Theorem 4.1. *The dynamic key sequence created by the scheme is secure if and only if function $f(.)$ is a one-way function.*

To prove the theorem, the three following lemmas are used.

Lemma 4.1. *If function $f(.)$ is a one-way function, the proposed scheme produces secure dynamic key sequences.*

Proof. Assume that the discarded dynamic keys in the sequence $DK_1, \ldots, DK_{c-1}$ are compromised. The probability of correctly guessing these keys is 1. Or, expressed another way:

$$Pr(Al(.) = DK_i) = 1, \forall i \in \{1 \ldots c-1\}. \tag{4.7}$$

From the equation (4.6) to generate the dynamic key:

$$DK_c = f(SK \oplus DK_{c-m} \oplus \ldots \oplus DK_{c-3} \oplus DK_{c-2} \oplus DK_{c-1}) \tag{4.8}$$

The function $f(.)$ can be also assumed to be known by everyone. To break the dynamic key generation scheme, SK is the only parameter of the function $f(.)$ that the adversary does not know. The chance of correctly guessing the current dynamic key DK_c can therefore be rewritten as:

$$Pr(Al(\{DK_i\}) = DK_c) = Pr(Al(.) = SK)). \tag{4.9}$$

By assuming the key exchange in step 1 and step 2 is secure, the temporary keys $TK_1, \ldots, TK_m$ in equation (4.3) are secure under cryptanalysis attack. Hence, the creation of SK in step 3 makes guessing SK also infeasible from the equation (4.4). The final method to compute SK is by calculating it from compromised discarded keys through an equation in (4.6), such as:

$$DK_{c-1} = f(SK \oplus DK_{c-m-1} \oplus \ldots \oplus DK_{c-3} \oplus DK_{c-2}). \tag{4.10}$$

Or:

$$\begin{aligned} f^{-1}(DK_{c-1}) &= f^{-1}(f(SK \oplus DK_{c-m-1} \oplus \ldots \oplus DK_{c-3} \oplus DK_{c-2})). \\ &= SK \oplus DK_{c-m-1} \oplus \ldots \oplus DK_{c-3} \oplus DK_{c-2}. \end{aligned} \tag{4.11}$$

Thus, SK can also be computed by:

$$SK = f^{-1}(DK_{c-1}) \oplus DK_{c-m-1} \oplus \ldots \oplus DK_{c-3} \oplus DK_{c-2}. \tag{4.12}$$

However, when $f(.)$ is a one-way function, it becomes infeasible to compute the reverse function of $f^{-1}(.)$; therefore:

$$Pr(Al(.) = SK)) \leq \frac{1}{2^s}. \tag{4.13}$$

Hence, we can deduce that:

$$Pr(Al(\{DK_i\}) = DK_c) < \frac{1}{2^s}. \tag{4.14}$$

In other words, it is infeasible to compute the current dynamic key DK_c from compromised discarded dynamic keys in the sequence; property 4.2 is therefore satisfied. □

Lemma 4.2. *If there is a feasible way to compute a reverse function $f^{-1}(.)$ of the function $f(.)$, it is possible to guess the next dynamic key from the compromised discarded keys.*

Proof. As in the proof in lemma 4.1, SK can be computed using the reverse function $f^{-1}(.)$ of function $f(.)$ in equation (4.12):

$$SK = f^{-1}(DK_{c-1}) \oplus DK_{c-m-1} \oplus \ldots \oplus DK_{c-3} \oplus DK_{c-2}. \tag{4.15}$$

Hence, if the reverse function $f^{-1}(.)$ is feasible to compute, it is also feasible to compute SK. When m continuous dynamic keys $DK_{c-m}, \ldots, DK_{c-1}$ in the sequence are compromised, the next dynamic key can also be computed by the equation (4.6). In other words, when function $f(.)$ is not a one-way function, the products from the dynamic key generation scheme are not able to secure the dynamic key sequences. □

Lemma 4.3. *Dynamic key sequences produced by the dynamic key generation function are unique.*

Proof. Function $f(.)$ gives only one unique value from $m+1$ input parameters including SK and $parameter_1, \ldots, parameter_m$. Therefore, with the identical input, the produced dynamic keys of any party are duplicated. □

Based on the above three lemmas, theorem 4.1 is proved as follows:

Proof. From lemma 4.1 and lemma 4.2, we can deduce that only one-way functions can be used to create secure dynamic keys. Lemma 4.3 verifies that the dynamic key generation scheme is able to produce unique dynamic key sequences. Based on these three lemmas, the conditions in property 4.1 and property 4.2 are satisfied; therefore, the dynamic key generation schemes using one-way functions can produce secure dynamic key sequences. □

By replacing a one-way hash function $h(.)$ with the function $f(.)$, the function $f(.)$ in equation (4.5) and (4.6) is rewritten as follows:

$$f(SK \oplus param_1 \oplus \ldots \oplus param_m) = h(SK \oplus param_1 \oplus \ldots \oplus param_m). \quad (4.16)$$

For implementation, any simple and fast one-way hash function can be used to generate sequences of dynamic keys. Most of the one-way functions based on the idea of public keys require more computational power than simple message hash functions such as MD5 and SHA. Although MD5 and SHA are efficient, they have a fixed output size that is not suitable for producing dynamic keys. In our opinion, HAVAL, which can produce different output sizes, is the most suitable one-way function for function $f(.)$.

The following corollary is used to prove that dynamic keys can be used to secure authentication protocols.

Corollary 4.1. *Without knowing $TK_1, \ldots TK_m$ and IK, it is infeasible to compute the dynamic keys DK_1 and DK_2.*

Proof. From theorem 4.1 and equation (4.5), without knowledge of $TK_1, \ldots, TK_m$ and SK, it is infeasible to compute DK_1 and so on DK_2. However, because of the equation (4.4), the possibility of guessing SK is equivalent to the possibility of guessing the sum of $TK_1 \ldots TK_m$ and IK:

$$Pr(Al(.) = SK)) = Pr(Al(.) = IK + TK_1 + \ldots + TK_m). \tag{4.17}$$

Hence, we can conclude that it is infeasible to compute DK_1 and DK_2 without knowing $TK_1, \ldots, TK_m$ and IK. □

Repeated Dynamic Key Generation

Once the sequence of n dynamic keys has been consumed, a new sequence of dynamic keys is generated by the repeated dynamic key regeneration scheme. The repeated dynamic key generation scheme has four steps (shown in figure 4.2):

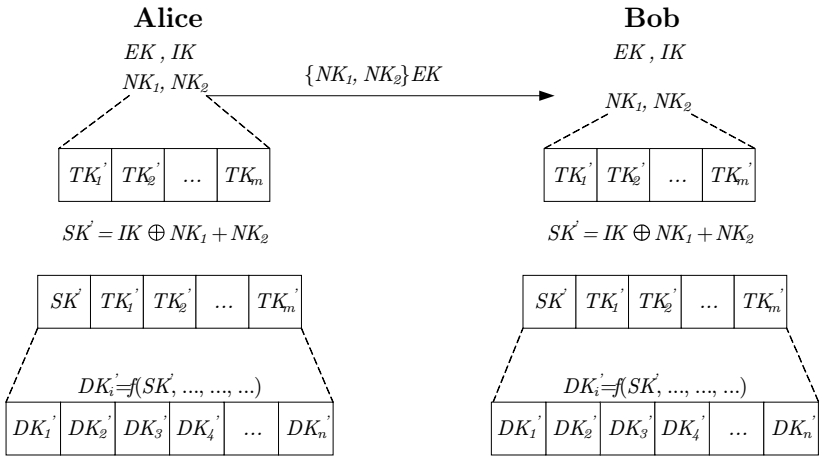


Figure 4.2: Repeated Dynamic Key Generation Scheme

Step 1: Both Alice and Bob calculate two new initial keys $EK^{'}$ and $IK^{'}$ from the last dynamic keys in the old sequence using a one-way hash function $h(.)$:

$$\begin{aligned} EK^{'} &= h(DK_{n-1} \oplus IK) \\ IK^{'} &= h(DK_n \oplus EK) \end{aligned} \tag{4.18}$$

Step 2: Alice and Bob need a new set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ to generate a new sequence of dynamic keys. To create the new set of temporary keys, Alice randomly generates two keys NK_1 and NK_2 and sends these, encrypted by $EK^{'}$, to Bob:

$$A \rightarrow B : \{NK_1, NK_2\}EK^{'}, h(NK_1 \oplus NK_2 \oplus EK^{'}) \tag{4.19}$$

Both Alice and Bob use these keys to compute the set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ as follows:

$$\begin{aligned} TK_1^{'} &= h(DK_{n-m+2} \oplus NK_1) \\ TK_2^{'} &= h(DK_{n-m+3} \oplus NK_1) \\ &\ldots \\ TK_{m-1}^{'} &= h(DK_n \oplus NK_1) \\ TK_m^{'} &= h(NK_1 \oplus NK_2) \end{aligned} \tag{4.20}$$

Step 3: Both Alice and Bob compute a new seed key $SK^{'}$ from $IK^{'}, NK_1$ and NK_2 using a bit-wise exclusive-OR and an addition operation:

$$SK^{'} = IK^{'} \oplus NK_1 + NK_2 \tag{4.21}$$

Steps 4: Both Bob and Alice generate the sequence of dynamic keys.

This step is the same as step 4 in the initial dynamic key generation scheme. The new seed key $SK^{'}$ and the new set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ are used to calculate a new sequence of dynamic keys $DK_1^{'}, \ldots, DK_m^{'}$.

$$\begin{aligned} DK_1^{'} &= f(SK^{'} \oplus TK_1^{'} \oplus \ldots \oplus TK_{m-2}^{'} \oplus TK_{m-1}^{'} \oplus TK_m^{'}) \\ DK_2^{'} &= f(SK^{'} \oplus TK_2^{'} \oplus \ldots \oplus TK_{m-1}^{'} \oplus TK_m^{'} \oplus DK_1^{'}) \\ &\ldots \\ DK_n^{'} &= f(SK^{'} \oplus DK_{n-m}^{'} \oplus \ldots \oplus DK_{n-3}^{'} \oplus DK_{n-2}^{'} \oplus DK_{n-1}^{'}). \end{aligned} \tag{4.22}$$

The repeated dynamic key generation scheme is used to generate a new sequence of dynamic keys when the old sequence is consumed. Each sequence of dynamic keys normally has a limited number of dynamic keys due to security issues. The maximum number of dynamic keys in a sequence is discussed in section 5.3.2. This ensures that the sequence of dynamic keys is renewed before the whole sequence becomes vulnerable to cryptanalysis attacks. In the next repeated dynamic key generation process, EK and IK are discarded. They are replaced by $EK^{'}$ and $IK^{'}$. Thus, even if a discarded sequence of dynamic keys were to be compromised, the new sequence created by the repeated dynamic generation scheme is secure.

4.2 The Group Key Manager for Wireless Networks

In this section, based on the hybrid group key management structure described in chapter 2, we develop a membership-oriented group management scheme that provides a secure group communication mechanism for managing user groups and service groups in wireless networks. The original idea of the membership-oriented group key management scheme was first proposed in [Wan09]. We developed and improved its performance in [NWL+09]. This group key management scheme supports users and services with multiple group memberships. By grouping the rekeying operations of multiple group members, rekeying operations for multiple membership users and services can be performed more efficiently. Before continuing this discussion, we point out that when discussing group key management, we do not use the term "user" as in other group key management schemes because in wireless networks, both users and services can be "users". One user can be a member of a user group and one service can also be a member of a service group. Instead, we replace the term "user" with the term "entity". In wireless networks, an entity can be either a user or a service. Where $\mathbb{E} = \mathbb{U} \cup \mathbb{S}$ is denoted as a set of all users and services or set of entities, $\mathbb{K}$ is denoted

as a set of keys and $\mathbb{R} \subset \mathbb{E} \times \mathbb{K}$ is denoted as an entity-key relationship. The group key management definition in equation (2.3) is re-defined as the trio $\{\mathbb{E}, \mathbb{K}, \mathbb{R}\}$.

4.2.1 Membership-Oriented Group Key Management Structure for Wireless Networks

The structure of membership-oriented group key management (MOGKM) [NWL+09] is divided into three different layers: the entity layer, the key-group layer and the detail layer. In the entity layer, the structure simply contains entity-groups (that is, service groups and user groups). In the key-group layer, the structure contains key-groups that support entities having multiple memberships. In the detail layer, each key-group is treated as an independent group. The detail layer is divided into multiple sub-groups to locally optimise the performance of individual rekeying operations for this key-group in wireless networks.

The Entity Layer

In the authentication model, entities (users/services) are grouped into entity-groups (user groups/service groups) or EGs. The set of all EGs is written as $\{EG\}$. Entities in an entity-group share a key named entity-group-key for securing group communication among group members in the entity-group. In this layer, there is no optimisation for group key management operations. All groups are directly derived from the user groups and service groups. Figure 4.3 illustrates a group key having three entity-groups: A, B and C. This layer is used to set up the key-group structure.

The Key-Group Layer

Entities with the same membership(s) are grouped in a key-group for the purpose of key management. In this layer, entities are separated into key-groups based on their membership(s). Each key-group has its own key named key-group-key so that

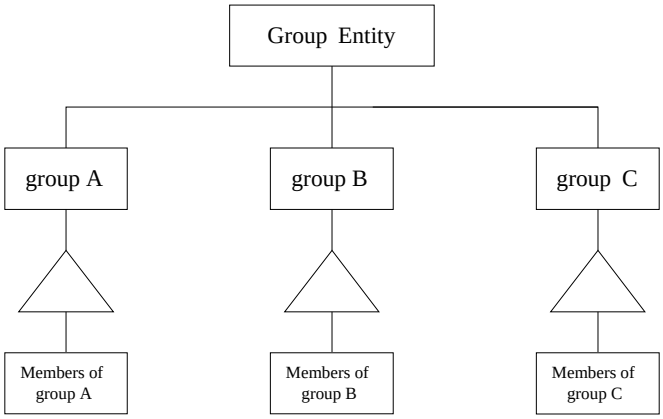


Figure 4.3: The Entity Layer

rekeying operations in the key-group can be handled securely and efficiently. The number of key-groups is determined by the number of combinations offered by service providers. The maximum possible number of key-groups from n entity-groups is $\sum_{i=1}^{n} C_n^i$. In reality, however, the number of key-groups may be smaller. Based on the membership, key-groups are classified into two types:

- single-membership key-groups that only contain entities having single membership; and
- multiple-membership key-groups that contain entities having two or more memberships.

A group KG is a key-group of entities having m membership(s) of $EG_1, \ldots, EG_m$, when all entities e in KG have m memberships of entity-group $EG_1, \ldots, EG_m$. Let the function to extract the set of entities groups from KG be $AEGS(KG)$. The formalisation to assign entity e into the key-group KG is then specified as follows:

$$\begin{aligned} & e \in KG, iff \\ & (i). \forall EG \in AEGS(KG), e \in EG \text{ and} \\ & (ii). \forall EG' \in \{EG\} \wedge EG' \notin AEGS(KG), e \notin EG'. \end{aligned} \tag{4.23}$$

The key-groups in the group key form a hierarchical structure. The hierarchical structure is divided into three levels according to the two different types of key-groups. The height of its tree structure is three, corresponding to its three levels. The root node is in level one. Below this, the second level contains single membership key-groups. The third level contains multiple membership key-groups. In the example in figure 4.4, the key-group KG_A in the second level is the key-group of entities having only one membership of entity-group EG_A. In the third level, another key-group, KG_{AB}, is recognised as a "sub-group" of the key-group KG_A if all entities in KG_{AB} also have membership of EG_A.

Each key-group possesses a key-group-key and entity-group-keys. Because key-groups in the second level of the hierarchical structure are single-membership key-groups, they are also recognised as entity-groups. Thus, each member of a single-membership key-group has one key being the key-group-key as well as the entity-group-key. The third level of the hierarchical structure contains multiple-membership key-groups. Each member of a key-group in this level obtains one key-group-key from the key-group and multiple entity-group-keys derived from its entity-groups memberships.

Figure 4.4 shows an example of a hierarchical structure of key-groups. The second level has three key-groups, A, B and C, for entities having a single membership with entity-groups A, B and C respectively. The third level contains key-groups of entities having either two or three memberships. These are key-groups A-B, A-C, B-C and A-B-C according to the key-group of entities having memberships of two entity-groups (A and B), (A and C) and (B and C) and entities having memberships of all three entity-groups A, B and C.

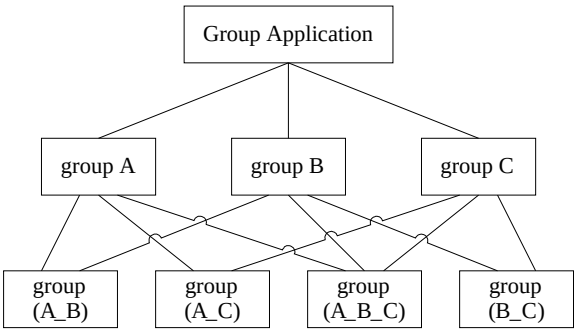


Figure 4.4: The Key-Group Layer

The Detail Layer

In the detail layer, each key-group is considered an independent group in order to optimise the rekeying operations for this group in wireless networks. To optimise rekeying operations in this key-group, the hybrid group key management scheme [WLS07] described in chapter 2 is employed. The hybrid group key management scheme in the realisation is managed by one centralised group key server (GKS) and $z + 1$ cell key servers (CKS). Rekeying operations in the detail layer are distributed in the hybrid group key management scheme via these key servers.

In the hybrid group key management scheme, entities of a key-group are categorised into $z + 1$ clusters. One cluster is named the ”leader-cluster” and the others are named ”member-clusters”. The group key shared by all entities in a cluster is called the ”cluster-key”. Each member-cluster contains entities either in local or adjacent wireless networks so that they can have efficient group communication using multi-casting. The leader-cluster includes z entities selected to be ”cluster-leaders” and the rest are called ”leader candidates”. The leader-cluster connects to z member-clusters via z cluster-leaders respectively. In each connection to a member-cluster, a cluster-leader has a cluster-key to the connected member-cluster. The leader-cluster and member-clusters divide the key-group in the detail layer into two logical levels (shown in figure 4.5).

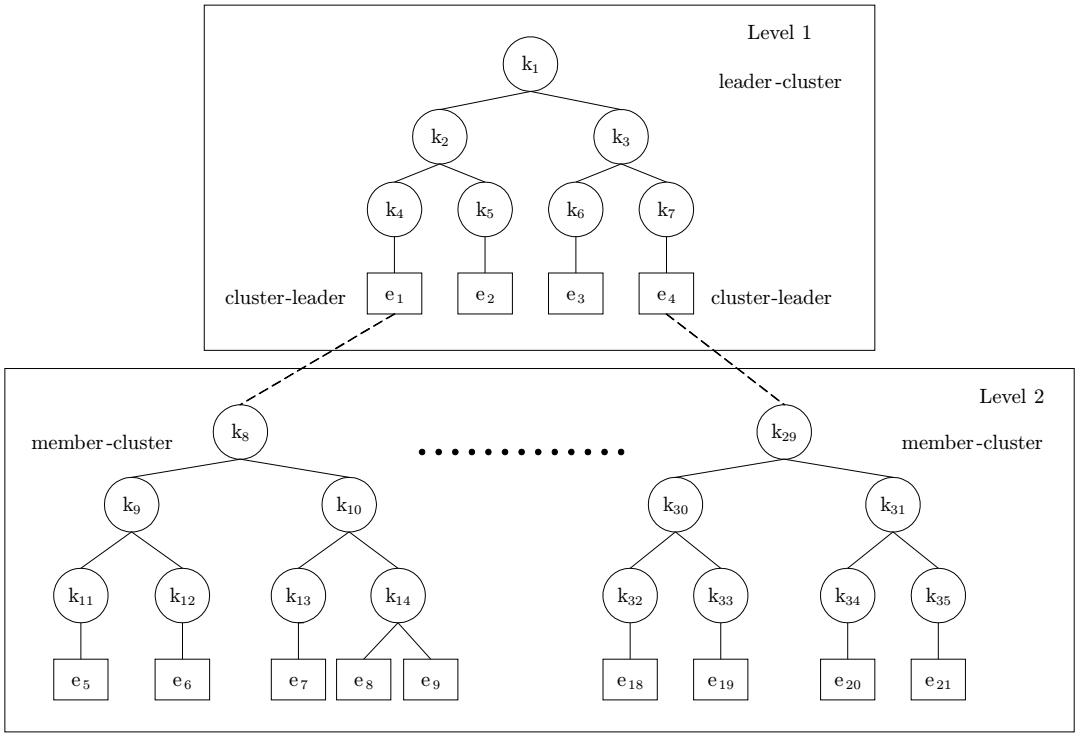


Figure 4.5: The Structure of a Key-Group in the Detail Layer

A key-group in the detail layer is separated into two levels. Level 1 contains the leader-cluster; level 2 contains member-clusters. The separation matches the two different levels in the wireless network architectures of hybrid group key management. By dividing the key-group into two levels, the effect of rekeying operations in a cluster is localised within one level.

Two stages are used to form the structure of the detail layer. In the first stage, the key tree of the leader-cluster of level 1 is formed. The size and the height of the key tree for the leader-cluster are set in the cluster policy. As entities join the detail layer, they are assigned into the key tree of the leader level until the leader-cluster is full. The first z entities become cluster-leaders while the others become leader-candidates. The second stage starts when the leader-cluster is full. In the second stage, member-clusters are formed and newly joined group entities are assigned into the member-cluster corresponding to their wireless network cell key servers.

Having explained the structure of MOGKM with multiple layers and levels of group key management, we now present rekeying operations.

4.2.2 Rekeying operations

In this section, key management operations in MOGKM are discussed. Because join and leave requests from the same entities are combined into new rekeying operations, MOGKM has four different types of rekeying operations: member join, member leave, key-group switch and handoff. Before describing the rekeying operations, commonly used notation in MOGKM are specified in table 4.2.2.

Table 4.2: Membership-Oriented Group Key Management Notation

e	the joining/leaving/switch entity	r	the number of entity memberships of e
GKS	the centralised group key server	CKS	a local cell group key server
h_l	the height of the leader-cluster	h_m	the height of the member-cluster
d	the degree of the key tree	KG_e	the key-group of e
EG_i	an entity-group	K_{KG_e}	the key-group-key of key-group KG_e
$SG_{l(i)}$	a sub-group level i of the leader-cluster	$K_{SG_{l(i)}}$	the group key of $SG_{l(i)}$
$SG_{m(i)}$	a sub-group level i in the member-cluster	$K_{SG_{m(i)}}$	the group key of $SG_{m(i)}$
es_i	an entity being in the same leaf node of e	DK_{es_i}	the current dynamic key of es_i
$SG_{m(i+1)(j)}$	a sub-group of $SG_{m(i)}$	$K_{SG_{m(i+1)(j)}}$	the group key of $SG_{m(i+1)(j)}$
$SG_{l(i+1)(j)}$	a sub-group of $SG_{l(i)}$	$K_{SG_{l(i+1)(j)}}$	the group key of $SG_{l(i+1)(j)}$
e_{leader_i}	a cluster-leader	K_{EG_i}	the entity-group-key of EG_i
$e_{newleader}$	an entity selected to be a new cluster-leader	DK_i^e	the current individual dynamic key of e
w_i	the number of affected key-groups from the rekeying of EG_i	$KG_{i(j)}$	an affected key-group from rekeying of EG_i
$K_{KG_{i(j)}}$	the key-group-key of $KG_{i(j)}$	er_i	an entity in the same leaf node of $e_{newleader}$
$\rightarrow$	sends a uni-cast message	$\Rightarrow$	sends a multi-cast message
z	the number of member-clusters	$K_{cluster_{(i)}}$	the cluster-key of $cluster_i$

A. Rekeying for Member Join

In MOGKM, a member join request is made by a non-membership entity that wants to be a member of one or more entity-group(s). Simultaneous requests to join multiple entity-groups of an entity e are combined into one member join request in MOGKM. Consequently, when the GKS receives a member join request, it determines the key-group for the joining entity based on the set of entity-group memberships. The relevant keys, including a key-group-key, entity-group-keys and sub-group's keys, are

required to be updated in order to ensure backward secrecy for the related groups. In MOGKM, the join process can be accomplished with the three following steps: (1) updating keys in the detail layer; (2) updating keys in the key-group layer; and (3) distributing keys to the entity.

Step 1: Updating keys in the detail layer.

Assume an entity e wants to join to r ($r \in \mathbb{N}, r > 0$) entity-groups $EG_1, \ldots, EG_r$. To begin, e sends a join request to the group key server (GKS). Before joining the key-group KG_e, e has to authenticate with the key server by sending a hash value of the result of its current dynamic key DK_i^e exclusive-OR with a nonce N_1. GKS responses to the authentication request by sending back N_1 encrypted by the dynamic key DK_{i+1}^e. GKS also sends another nonce N_2 to challenge e. e sends back N_2 encrypted by its dynamic key DK_{i+2}^e as the response to complete the authentication step. Once successfully authenticated to the GKS, e is assigned to the key-group KG_e according to its r membership(s) of entity-groups $EG_1, \ldots, EG_r$. In order to enforce backward secrecy, the GKS notifies local cell key server (CKS) to rekey key-group KG_e.

1. $e \rightarrow GKS$: join request, $e, EG_1, \ldots EG_r, N_1, h(N_1 \oplus DK_i^e)$.
2. $GKS \rightarrow e$: $N_2, \{N_1\}DK_{i+1}^e$.
3. $e \rightarrow GKS$: $\{N_2\}DK_{i+2}^e$.
4. $GKS \Rightarrow$ all $CKSs$: {member join rekeying for key-group KG_e}.

In the detail layer, the next step - that is, the assignment of e - depends on the status of the key tree. There are two possible scenarios. If the key tree in the leader-cluster is not full, the entity e is inserted into the leader-cluster (a). Otherwise, the CKS assigns e to a member-cluster (b).

(a). If e is assigned into the leader-cluster, the rekeying in the detail layer is locally performed by the CKS in the leader cluster. To do the rekeying, the CKS controlling the leader-cluster generates a new key-group-key K'_{KG_e} for the key-group KG_e and

related sub-groups' keys. Let the height of the leader-cluster of the key-group KG_e be h_l. Let $SG_{l(h_l)}$ be the sub-group at the leaf node where e is going to be assigned in the key tree of the leader-cluster. Let $SG_{l(1)}, \ldots, SG_{l(h_l-1)}$ be sub-groups on the path from the root to $SG_{l(h_l)}$. The CKS generates h_l new keys $K'_{SG_{l(1)}}, \ldots, K'_{SG_{l(h_l)}}$ to replace the old keys $K_{SG_{l(1)}}, \ldots, K_{SG_{l(h_l)}}$ of sub-groups $SG_l(1), \ldots, SG_l(h_1)$ respectively. The new keys are updated to all existing entities in the cell as follows:

5. $CKS \Rightarrow$ the leader-cluster : $\{K'_{KG_e}\}K_{KG_e}, \{K'_{SG_{l(1)}}\}K_{SG_{l(1)}}, \ldots, \{K'_{SG_{l(h_l)}}\}K_{SG_{l(h_l)}}$.

(b). If e is assigned into a member-cluster, the rekeying is locally performed by the CKS in the member cluster. Like the rekeying in the leader-cluster, the CKS controlling the member-cluster generates a new key-group-key K'_{KG_e} and related sub-groups' keys. Let the height of the member-cluster be h_m. Let $SG_{m(h_m)}$ be the sub-group at the leaf node where e is going to be assigned in the key tree of the member-cluster. Let $SG_{m(1)}, \ldots, SG_{m(h_m-1)}$ be sub-groups on the path from the root to $SG_{m(h_m)}$. The CKS updates h_m new sub-groups' keys $K'_{SG_{m(1)}}, \ldots, K'_{SG_{m(h_m)}}$ to replace old keys $K_{SG_{m(1)}}, \ldots, K_{SG_{m(h_m)}}$ of sub-groups $SG_{m(1)}, \ldots, SG_{m(h_m)}$ respectively. The rekeying message in the detail layer is specified as follows:

5. $CKS \Rightarrow$ the member-cluster : $\{K'_{KG_e}\}K_{KG_e}, \{K'_{SG_{m(1)}}\}K_{SG_{m(1)}}, \ldots, \{K'_{SG_{m(h_m)}}\}K_{SG_{m(h_m)}}$.

After rekeying the detail layer, the GKS rekeys the key-group layer, as follows.

Step 2: Updating keys in the key-group layer.

In step 2, the GKS determines and updates the affected keys that relate to the key-group KG_e. When e joins key-group KG_e, the affected keys are the entity-group keys of the entity-groups in $AEGS(KG_e)$. These entity-group-keys must be changed in order to ensure backward secrecy in the key-group layer. Let r affected entity-groups be $EG_1, \ldots, EG_r$. Let their entity-group-keys be $K_{EG_1} \ldots K_{EG_r}$. The GKS

generates r entity-group-keys $K^{'}_{EG_1}, \ldots, K^{'}_{EG_r}$ to replace old keys ($K_{EG_1} \ldots K_{EG_r}$). The entity-group-key rekeying message is specified as follows:

6. $GKS \Rightarrow$ all entity-groups : $\{K^{'}_{EG_1}\}K_{EG_1}, \ldots, \{K^{'}_{EG_r}\}K_{EG_r}$.

Step 3: Distributing keys to the entity.

After the rekeying is performed, e receives the new keys it is entitled to know. These keys include the new key-group-key, the new sub-groups' keys ($KG^{'}_{e}$ and either $K^{'}_{SG_{m(1)}}, \ldots, K^{'}_{SG_{m(h_m)}}$ or $K^{'}_{SG_{l(1)}}, \ldots, K^{'}_{SG_{l(h_l)}}$ from step 1) and the new keys of entity-groups ($K^{'}_{EG_1}, \ldots, K^{'}_{EG_r}$ from step 2). All of these are encrypted by the dynamic key DK^{e}_{i+1} of e. There are two possible scenarios:

(a). If e was to be assigned into the leader-cluster, the key distribution message for entity e would be specified as follows.

7. $GKS \rightarrow e : \{K^{'}_{SG_{l(1)}}, \ldots, K^{'}_{SG_{l(h_l)}}, K^{'}_{EG_1}, \ldots, K^{'}_{EG_r}, K^{'}_{KG_e}\}DK^{e}_{i+1}$.

(b). Otherwise, after e is assigned into the member-cluster, e receives the following message from the GKS:

7. $GKS \rightarrow e : \{K^{'}_{SG_{m(1)}}, \ldots, K^{'}_{SG_{m(h_m)}}, K^{'}_{EG_1}, \ldots, K^{'}_{EG_r}, K^{'}_{KG_e}\}DK^{e}_{i+1}$.

B. Rekeying for Member Leave

In MOGKM, a leave operation occurs when a member e wants to exit all the entity-groups to which it subscribes. (Should an entity want to remain a member of some groups, a switch operation, rather than a leave operation, is required. This situation is described in the next section). The leave operation can be either voluntary (when the entity makes the decision to leave all entity-groups) or involuntary (when the entity is evicted from all entity-groups by the GKS). The leave operation can be considered a combination of multiple simultaneous member leave requests by an entity. Rekeying must be performed to ensure forward secrecy. Three steps are involved in the rekeying

for a leave operation: (1) de-registering the member; (2) updating keys in the detail layer; and (3) updating keys in the key-group layer.

Step 1: De-registering the member.

In this step, an entity e submits a leave request to the GKS to leave all groups:

1. $e \rightarrow GKS$: {leave request}.
2. $GKS \Rightarrow$ all $CKSs$: {rekey for leaving of $e, \{K'_{KG_e}\}DK_i^{CKS}$}.

After receiving this leave request, the GKS deletes e's membership from the corresponding group member list.

Step 2: Updating keys in the detail layer.

After removing e from the key-group, to protect forward secrecy, the GKS updates the related keys of sub-groups and the key-group KG_e in the detail layer. The original location of entity e - either in the leader-cluster or in a member-cluster - affects the rekeying process, as does the composition of the cluster. Consequently, there are four situations that require different rekeying treatments: (a) the leaving entity e is from a member-cluster; (b) the leaving entity e is a leader-candidate from a leader-cluster; (c) the leaving entity e is a cluster-leader and no replacement leader-candidate is available; and (d) the leaving entity e is a cluster-leader and no leader-candidate is available.

(a). When e is originally in a member-cluster, the rekeying is performed locally within the member-cluster. Related sub-groups are groups having membership of e in the member-cluster. Let the related sub-groups on the path from the root to the leaf node be $SG_{m(1)}, \ldots, SG_{m(h_m)}$ and let their keys be $K_{SG_{m(1)}}, \ldots, K_{SG_{m(h_m)}}$ respectively. In order to update these keys, the CKS generates new keys $K'_{SG_{m(1)}}, \ldots, K'_{SG_{m(h_m)}}$ and updates them to related sub-groups. Let $d(d \in \mathbb{N}, d > 0)$ be the degree of the key tree in the member-cluster. Let $es_1, \ldots, es_{d-1}$ be entities in the same leaf node as e and let $DK_{es_1}, \ldots, DK_{es_{d-1}}$ be their dynamic keys. Let $SG_{m(i)(1)}, \ldots, SG_{m(i)(d)}, i \in \mathbb{N}, 0 \leq$

$i \leq h_m - 1$ be sub-groups of $SG_{m(i)}$ and let $K_{SG_{m(i)(1)}}, \ldots, K_{SG_{m(i)(d)}}$ be their keys. The procedure to update the keys is performed in a bottom-up order, as follows:

3. $CKS \rightarrow es_1 : \{K'_{SG_{m(h_m)}}\}DK_i^{es_1}$.

   ...

$d+1$. $CKS \rightarrow es_{d-1} : \{K'_{SG_{m(h_m)}}\}DK_i^{es_{d-1}}$.

$d+2$. $CKS \Rightarrow$ the member-cluster : $\{K'_{SG_{m(h_m-1)}}\}K_{SG_{m(h_m)(1)}} \cdots$
$\{K'_{SG_{m(h_m-1)}}\}K_{SG_{m(h_m)(d)}}$.

...

h_m+d. $CKS \Rightarrow$ the member-cluster: $\{K'_{SG_{m(1)}}\}K_{SG_{m(2)(1)}}, \ldots \{K'_{SG_{m(1)}}\}K_{SG_{m(2)(d)}}$.

After the rekeying, the cluster-key (the key of the root of the member-cluster in the detail level) has a new value, $K'_{SG_{m(1)}}$. The GKS updates this new key to the cluster-leader of the member-cluster denoted as e_{leader}.

h_m+d+1. $CKS \rightarrow e_{leader} : \{K'_{SG_{m(1)}}\}DK_i^{e_{leader}}$.

Finally, the key-group-key of KG_e is rekeyed as follows:

h_m+d+2. $CKS \Rightarrow$ the member-cluster : $\{K'_{KG_e}\}K'_{SG_{m(1)}}$.

h_m+d+3. $CKS \Rightarrow$ the leader-cluster : $\{K'_{KG_e}\}K_{SG_{l(1)}}$.

If e was originally in the leader-cluster, the rekeying is more complicated. Three different scenarios relate to the three different roles of e in the leader-cluster: (i) e is a leader-candidate; (ii) e is a cluster-leader and a leader candidate is available; or (iii) e is a cluster-leader and no leader-candidate is available.

(b). In the first scenario, when e is only a leader candidate, the rekeying is performed locally within the leader-cluster. Related sub-groups are groups having membership of e in the leader-cluster. Let the related sub-groups on the path from the root to the leaf node be $SG_{l(1)}, \ldots, SG_{l(h_l)}$ and let their keys be $K_{SG_{l(1)}}, \ldots, K_{SG_{l(h_l)}}$ respectively. In order to update these keys, the CKS generates new keys $K'_{SG_{l(1)}}, \ldots, K'_{SG_{l(h_l)}}$ and updates them to related sub-groups. Let $es_1, \ldots, es_{d-1}$ be entities in the same

leaf node as e and let $DK_i^{es_1}, \ldots, DK_i^{es_{d-1}}$ be their dynamic keys. Let $SG_{l(i)(1)}, \ldots,$ $SG_{l(i)(d)}, i \in \mathbb{N}, 0 \leq i \leq h_l - 1$ be sub-groups of $SG_{l(i)}$ and let $K_{SG_{l(i)(1)}}, \ldots, K_{SG_{l(i)(d)}}$ be their keys. The procedure to update the keys is performed in a bottom-up order, as follows:

3. $CKS \rightarrow es_1 : \{K'_{SG_{l(h_l)}}\}DK_i^{es_1}$.

$\ldots$

$d+1$. $CKS \rightarrow es_{d-1} : \{K'_{SG_{l(h_l)}}\}DK_i^{es_{d-1}}$.

$d+2$. $CKS \Rightarrow$ the leader-cluster : $\{K'_{SG_{l(h_l-1)}}\}K_{SG_{l(h_l)(1)}} \ldots \{K'_{SG_{l(h_l-1)}}\}K_{SG_{l(h_l)(d)}}$.

$\ldots$

$h_l + d$. $CKS \Rightarrow$ the leader-cluster : $\{K'_{SG_{l(1)}}\}K_{SG_{l(2)(1)}}, \ldots \{K'_{SG_{l(1)}}\}K_{SG_{l(2)(d)}}$.

h_l+d+1. $CKS \Rightarrow$ the leader-cluster : $\{K'_{KG_e}\}K'_{SG_{l(1)}}$.

Finally, the cluster-leaders multi-cast the new key-group-key to their member-clusters. Let the cluster-leaders in the detail layer be $e_{leader_1}, \ldots, e_{leader_z}$ and let $K_{cluster_{(1)}}$, $\ldots, K_{cluster(z)}$ be the cluster-key of the member-clusters corresponding to $e_{leader_1}, \ldots,$ e_{leader_z}. The key-group-key update messages are written as follows:

$h_l + d + 2$. $e_{leader_1} \Rightarrow$ the member-cluster$_1$: $\{K'_{KG_e}\}K_{cluster_{(1)}}$.

$\ldots$

$h_l+d+z+1$. $e_{leader_z} \Rightarrow$ the member-cluster$_z$: $\{K'_{KG_e}\}K_{cluster_{(z)}}$.

(c). When e is a cluster-leader and at least one leader-candidate is available, the rekeying procedure in (b) is performed and one of the leader-candidates is promoted to be cluster-leader. In addition to the rekeying messages in (b), there are extra messages when the leader-candidate, $e_{newleader}$, is promoted to cluster-leader. The extra messages contain the new cluster-key for the new cluster-leader to control its cluster. The rekeying process begins with rekeying messages for the leaving member in the leader-cluster, as follows:

3. $CKS \rightarrow es_1 : \{K'_{SG_{l(h_l)}}\}DK_i^{es_1}$.

...

$d+1.$ $CKS \rightarrow es_{d-1} : \{K'_{SG_{l(h_l)}}\}DK_i^{es_{d-1}}$.

$d+2.$ $CKS \Rightarrow$ the leader-cluster : $\{K'_{SG_{l(h_l-1)}}\}K_{SG_{l(h_l)(1)}} \ldots \{K'_{SG_{l(h_l-1)}}\}K_{SG_{l(h_l)(d)}}$.

...

$h_l+d.$ $CKS \Rightarrow$ the leader-cluster : $\{K'_{SG_{l(1)}}\}K_{SG_{l(2)(1)}}, \ldots \{K'_{SG_{l(1)}}\}K_{SG_{l(2)(d)}}$.

The CKS then updates the new cluster-key, $K'_{SG_{m(1)}}$, for the member-cluster:

$h_l+d+1.$ $CKS \Rightarrow$ the member-cluster:$\{K'_{SG_{m(1)}}\}K_{SG_{m(2)(1)}}, \ldots, \{K'_{SG_{m(1)}}\}K_{SG_{m(2)(d)}}$.

and promotes the leader candidate $e_{newleader}$ to be the new leader.

$h_l+d+2.$ $CKS \rightarrow e_{newleader} : \{K'_{SG_{m(1)}}\}DK_i^{e_{newleader}}$.

Finally, the CKS sends the new key-group-key K'_{KG_e} to everyone in the leader-cluster and in turn each cluster-leader $e_{leader_1}, \ldots, e_{leader_z}$ sends this new key-group-key to members in its member cluster. (Note that $e \notin \{e_{leader_1}, \ldots, e_{leader_z}\}, e_{newleader} \in \{e_{leader_1}, \ldots, e_{leader_z}\}$):

$h_l+d+3.$ $CKS \Rightarrow$ the leader-cluster : $\{K'_{KG_e}\}K'_{SG_{l(1)}}$.

$h_l+d+4.$ $e_{leader_1} \Rightarrow$ the member-cluster$_1$: $\{K'_{KG_e}\}K_{cluster_{(1)}}$.

...

$h_l+d+z+3.$ $e_{leader_z} \Rightarrow$ the member-cluster$_z$: $\{K'_{KG_e}\}K_{cluster_{(z)}}$.

(d). When e cannot find any leader candidate, the CKS selects a new leader, $e_{newleader}$, from the member-cluster. To perform the rekeying for this scenario, the CKS updates related sub-group keys for the departure of $e_{newleader}$ in the member-cluster:

3. $CKS \rightarrow es_1 : \{K'_{SG_{m(h_m)}}\}DK_i^{es_1}$.

...

$d+1.$ $CKS \rightarrow es_{d-1} : \{K'_{SG_{m(h_m)}}\}DK_i^{es_{d-1}}$.

$d+2$. $CKS \Rightarrow$ the member-cluster: $\{K'_{SG_{m(h_m-1)}}\}K_{SG_{m(h_m)(1)}}, \ldots$

$\{K'_{SG_{m(h_m-1)}}\}K_{SG_{m(h_m)(d)}}$.

$\ldots$

h_m+d. $CKS \Rightarrow$ the member-cluster: $\{K'_{SG_{m(1)}}\}K_{SG_{m(2)(1)}}, \ldots \{K'_{SG_{m(1)}}\}K_{SG_{m(2)(d)}}$.

The CKS also rekeys the related sub-group keys for the leaving e from the leader-cluster:

h_m+d+1. $CKS \rightarrow er_1 : \{K'_{SG_{l(h_l)}}\}DK_i^{er_1}$.

$\ldots$

h_m+2d-1. $CKS \rightarrow er_{d-1} : \{K'_{SG_{l(h_l)}}\}DK_i^{er_{d-1}}$.

h_m+2d. $CKS \Rightarrow$ the leader-cluster: $\{K'_{SG_{l(h_l-1)}}\}K_{SG_{l(h_l)(1)}} \ldots \{K'_{SG_{l(h_l-1)}}\}K_{SG_{l(h_l)(d)}}$.

$\ldots$

h_m+2d+h_l-2. $CKS \Rightarrow$ the leader-cluster:$\{K'_{SG_{l(1)}}\}K_{SG_{(2)(1)}}, \ldots \{K'_{SG_{l(1)}}\}K_{SG_{l(2)(d)}}$.

$e_{newleader}$ is promoted into the position of cluster-leader in the leader-cluster. After that, the CKS sends the updated key-group-key for the key-group K'_{KG_e} to members of the leader-cluster:

h_m+2d+h_l-1. $CKS \rightarrow e_{newleader} : \{K'_{SG_{m(1)}}, K'_{SG_{l(1)}}, \ldots, K'_{SG_{l(h_l)}}, \}DK_i^{e_{newleader}}$.

h_m+2d+h_l. $CKS \Rightarrow$ the leader-cluster : $\{K'_{KG_e}\}K'_{SG_{l(1)}}$.

In turn, cluster-leaders send the new key-group-key K'_{KG_e} to their member-clusters:

h_m+2d+h_l+1. $e_{leader_1} \Rightarrow$ the member-cluster$_1$: $\{K'_{KG_e}\}K_{cluster_{(1)}}$.

$\ldots$

h_m+2d+h_l+z. $e_{leader_z} \Rightarrow$ the member-cluster$_z$: $\{K'_{KG_e}\}K_{cluster_{(z)}}$.

Step 3: Updating keys in the key-group layer.

The GKS determines and generates new supporting keys to replace affected entity-group-keys after the departure of e. When e leaves the key-group KG_e, the

directly affected key-group is the key-group KG_e. Let KG_e be the subgroup of entity-groups $EG_1, \ldots, EG_r$ in the entity layer. The affected entity-groups in the entity layer are therefore $EG_1, \ldots, EG_r$. Let entity-group-keys of these affected entity-groups be $K_{EG_1}, \ldots, K_{EG_r}$, These keys must be changed in order to ensure forward secrecy in the entity layer. The GKS generates new values of the keys $K'_{EG_1}, \ldots, K'_{EG_r}$. These new keys are sent to the related key-groups in the key-group layer. Therefore, the indirectly affected key-groups are the key-groups that are sub-groups of the entity group $EG_1, \ldots, EG_r$. For each affected entity-group $EG_i, EG_i \in \{EG_1, \ldots, EG_r\}$, let w_i be the number of the indirectly affected key-groups $KG_{i(1)}, \ldots, KG_{i(w_i)}$. Let the indirectly affected keys of these key-groups be $K_{KG_{i(1)}}, \ldots, K_{KG_{i(w_i)}}$. The process of rekeying for both directly and indirectly affected group keys is specified as follows:

h_m+2d+h_l+z+1. $GKS \Rightarrow$ all key-groups : $\{K'_{EG_1}\}K_{KG_{1(1)}}, \ldots \{K'_{EG_1}\}K_{KG_{1(w_1)}}$.

...

h_m+2d+h_l+z+2. $GKS \Rightarrow$ all key-groups : $\{K'_{EG_r}\}K_{KG_{r(1)}}, \ldots \{K'_{EG_r}\}K_{KG_{r(wr)}}$.

C. Rekeying for Key-Group Switch

The previous discussion focused on member leave operations. Member leave operations occur when a member e wants to exist all the entity-groups to which it subscribes. However, should an existing member of a key-group exit or/and join other entity-groups, then a key-group switch operation is required. For example, assume that e has subscribed to r entity-groups. If e only leaves p entity-group(s) ($p < r$), then a key-membership switch operation is used, because the entity is still a member of one or more group entity-group(s). A key-group switch request is equivalent to the combination of multiple normal member join and leave requests. Its rekeying can therefore be considered a combination of two rekeyings: a rekeying of a member leave from the old key-group and a rekeying of a member join to the new key-group.

When e makes a key-group switch request from key-group KG_O to key-group KG_F, the GKS rekeys the affected group keys from the list of ceased memberships and the list of new memberships. Let $OEG_1, \ldots, OEG_r$ be entity-group memberships of key-group KG_O and let $FEG_1, \ldots, FEG_s$ be entity-group memberships of key-group KG_F. By removing the memberships common to the two lists, the GKS determines the list of ceased memberships and the list of new memberships. Let the list of ceased entity-group memberships of e be $CEG_1, \ldots, CEG_p$ and the list of new entity-group memberships be $NEG_1, \ldots, CEG_q$ (note that $\{CEG_1, \ldots, CEG_p\} \cap \{CEG_1, \ldots, CEG_q\} = \varnothing$). The list of affected entity-group-keys from ceased memberships and new memberships is written as $K_{CEG_1}, \ldots, K_{CEG_p}$ and $K_{NEG_1}, \ldots, K_{NEG_q}$. To ensure both forward and backward secrecy in MOGKM, the rekeying is performed through the following three steps: (1) updating keys in the detail layer; (2) updating keys in the key-group layer; and (3) distributing the new keys to the entity.

Step 1 Updating keys in the detail layer.

In this step, e sends a request to the GKS:

1. $e \rightarrow GKS$: {group switch request, KG_O, KG_F}.

After receiving the request, the GKS updates memberships with the corresponding group member list. The request to move from the old key-group KG_O to KG_F is a combination of two requests: the first a request to leave key-group KG_O and the second a request to join key-group KG_F. Hence, the rekeying in this step comprises two parts: a) updating keys in the detail layer of the rekeying member leave operation (step 2 in the previous discussion B) in order to change KG_O and b) updating keys in the detail layer of the rekeying member join operation (step 1 in the previous discussion A) in order to change KG_F.

Step 2 Updating keys in the key-group layer.

Once memberships are updated, the entity-group-keys also need to be updated for forward and backward secrecy. GKS generates the new entity-group-keys K'_{CEG_1}, $\ldots K'_{CEG_p}$ and $K'_{NEG_1}, \ldots K'_{NEG_q}$ to ensure forward and backward secrecy. This rekeying step is another combination of updating keys in the key-group layer of the rekeying member leave operation (step 3 in the previous discussion B) and updating keys in the key-group layer of the rekeying member join operation (step 2 in the previous discussion A).

Step 3 Distributing the new group keys to the entity.

Finally, the GKS sends the new sub-group keys, key-group-keys and entity-group-keys to e.

3. $GKS \rightarrow e : \{K'_{SG_1}, \ldots, K'_{SG_h}, K'_{NEG_1}, \ldots, K'_{NEG_q}, K'_{KG_e}\}DK_i^e$.

The above discussion describes three rekeying operations that make MOGKM suitable for use in wireless networks. As MOGKM efficiently supports entities with multiple membership entities it can be used as a group manager to manage group memberships of user and service groups. A further benefit is that MOGKM's extension for wireless networks includes a handoff operation. This operation, described in the following section, provides a mechanism that enables mobile devices to move between different wireless networks.

D. The Handoff Operation

Because of their mobility, entities may experience frequent handoff while moving between different wireless networks. Handoff in wireless networks occurs when the wireless signal received from the current base station of the wireless network decreases while the wireless signal received from the new base station increases. During handoff, an entity disconnects from one wireless network base station and reconnects

to another. In the process, the connections of the entity are interrupted. When re-connecting in new wireless networks, the entity may have to re-authenticate before continuing its services. As handoff in group key management for wireless networks can be considered a combination of a member leave and a subsequent member join operation, handoff processes may involve substantial rekeying. When an entity experiences frequent handoff, rekeying operations may create major performance concerns for MOGKM.

Many approaches ([UD06] [KKRD03] [STL04]) can be used to reduce the overhead associated with the handoff problem and the associated rekeying operations. In this thesis, we adopt the approach described in [WLS07] and [Wan09] to solve the handoff issue in wireless networks. This approach relies on the handoff patterns investigated in [BCS+99] to improve rekeying performance in wireless cellular networks.

When an entity e detects that handoff is about to happen, it sends a handoff join request to the new CKS in the new base station. After receiving the handoff request, the new CKS validates the entity by asking the current CKS for member authentication. The communication messages among parties are written as follows:

1. $e \rightarrow$ new CKS : {handoff request,current CKS, e}.
2. new $CKS :\rightarrow$ current CKS : {handoff authentication, e}K_{CKS}.
3. current $CKS :\rightarrow$ new CKS : {authentication response}K_{CKS}.

If the authentication for the entity e from the current CKS is successful, the new CKS puts e into a handoff waiting list. When the handoff operation is completed, e is placed in the new CKS. The new CKS performs join rekeying for e by invoking step 1 in member join so that e can join the new group key. At the old CKS, step 2 in member leave is invoked in order to rekey e's leaving the key-group.

No entity-group membership changes in this operation. Rekeying is performed locally in the detail layer without involving the central GKS. The rekeying of the

old CKS and the rekeying of the new CKS can be performed simultaneously, thus minimising performance overhead.

4.3 Applying MOGKM to Realise the Group Manager of the AMUS Authentication Model

The group manager is realised by MOGKM. It has two main functions: to manage group memberships for user and service groups and to distribute authentication keys to authorised group members. An application of MOGKM is able to manage, securely and efficiently, user groups and service groups of members having multiple memberships. Authentication keys can also be distributed to users via secure group communications in MOGKM. MOGKM thus provides a framework that supports the basic functions of the group manager and secures group communication for wireless network users and services.

In addition to serving as the group manager, MOGKM in this model also supports group communications among members. Many services in wireless networks (such as emergency systems, tele-conferences, mobile IPTV [PJH08], online games [JSB07], Google WAVE group communication and Google WAVE collaboration [Par10]) utilise multi-casting for communicating with their users. Using multi-casting, the information from these services can be delivered efficiently to members of user groups. Similar to users, members of service groups can also obtain efficient communication among group members via multi-casting. However, even when using multi-cast communication among group members, a platform is required that can provide strong security. The following discussion demonstrates how MOGKM in wireless networks can also provide an advanced security platform for authentication and group communication among members of user groups and service groups.

To support authentication in the AMUS model, each group in MOGKM has two keys: an entity group key, K_{EG}^{entity}, and an authentication key, K_{EG}^{auth}. The entity group key is derived from the entity layer in MOGKM. The entity group can be either a user group or a service group. Communications among members of an entity group can be secured by the entity group key, K_{EG}^{entity}. However, this key is not used for authentication in the authentication layer. Instead, K_{EG}^{auth} is the authentication key for the group identity of group members. The process to distribute and update the authentication key among members of the entity group using multi-casting is secured by the entity group key, K_{EG}^{entity}. Any group membership change also invokes rekeying the two keys (K_{EG}^{entity} and K_{EG}^{auth}).

To protect the authentication key K_{EG}^{auth} from cryptanalysis attack, a periodic rekeying operation is added to MOGKM. This operation is independent of membership changes; no relationship exists between the periodic rekeying operation and the four original rekeying operations in MOGKM. The rekeying frequently refreshes the authentication keys and consequently reduces the risk of cryptanalysis attack. In general, users are more likely to change group memberships than service groups. Periodic rekeyings for user groups are thus less frequent than those for service groups because service groups have more frequent rekeying resulting from membership changes.

4.3.1 Periodic Rekeying

Because long-term authentication keys are susceptible to cryptanalysis attack risks, they need to be refreshed after a certain amount of time. With an adequate computation resource, an adversary is able to break any cryptography key in a specific amount of time. Hence, depending on the security requirement, the period between two rekeyings can be determined. If the security requirement for an authentication key for an entity group is a, the security requirement to secure the authentication key

from cryptanalysis attack risks is then written as:

$$Pr(Al(K_{EG}^{auth})) \leq a \tag{4.24}$$

If the key size of K_{EG}^{auth} is s bit length and within a unit of time, an adversary is able to perform z trials to break the cryptography of K_{EG}^{auth}. After T units of time, the adversary can conducts zT trials. Assuming that the adversary can only use brute force to break the cryptography, the equation (4.24) is rewritten as follows:

$$\frac{1}{2^s - zT} \leq a \tag{4.25}$$

Or:

$$T \leq \frac{2^s}{z} - \frac{1}{az} \tag{4.26}$$

After T units of time ($T \leq \frac{2^s}{z} - \frac{1}{az}$) without being rekeyed, the authentication key is refreshed by a periodic rekeying operation. This operation is invoked by GKS. The periodic rekeying is specified as follows:

1. $GKS \Rightarrow EG : \{K_{EG}^{auth'}\}K_{EG}^{entity}$.

4.3.2 The Group Manager Components and Memberships

As derived from the AMUS authentication model, the group manager has two group key management components: UGM and SGM. These two components can either be merged into one service or separated into two different services. Because of the differences in the security requirements of the two types of groups, in the realisation UGM and SGM are denoted as two separate services.

UGM and SGM are the two group key servers of MOGKM that manage user groups and service groups. In the model, user groups form a hierarchical structure. Membership changes in the user group hierarchical structure are controlled by the rekeying operations of UGM. Each membership of a user is correlated to a group

key and an authentication key. Users use their group keys to communicate with other group members and authentication keys to authenticate their group membership. Similar to user groups, service groups also form a hierarchical structure managed by SGM. Both of the two hierarchical key structures use the periodic rekeying operation to refresh the authentication keys. The details of how to use the authentication keys of user groups and service groups for authentication are described in the next section.

4.4 Applying Dynamic Key Cryptography to Realise the Authentication Controller of the AMUS Authentication Model

Each entity (service or user) in this authentication has an authentication library containing three modules: a dynamic key module, a group key management module and an authentication protocols module. The dynamic key module handles the process to generate dynamic keys from the dynamic key sequences. This is the basic module used by all components in the realisation. The group key management module handles communication with the group manager to update group authentication keys of the entity. The group key management module has a sequence of individual dynamic keys to authenticate and communicate with the group manager. Finally, the authentication protocols module handles authentication protocols and performs these protocols during the authentication. The authentication library is illustrated in figure 4.6.

The core of the authentication controller in the AMUS authentication model is the authentication service AS. Similar to other entities, AS also has an authentication library that receives authentication requests from users and services. In order to validate the claimed identities of authentication requests, AS is, by default, a member of all the user and service groups so that it knows the group authentication keys

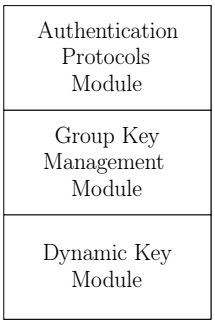


Figure 4.6: The Modules of an Authentication Library

of all these groups. Based on its knowledge of shared authentication keys, AS can validate identities and issue authentication tickets to authorised users and services via authentication protocols. The security of authentication within the realisation is heavily reliant upon the important role of AS and the authentication protocols.

Because existing authentication protocols do not achieve high levels of both security and efficiency in wireless networks, new authentication protocols are proposed for this realisation. Current authentication protocols, whether based on symmetric or asymmetric long-term authentication keys, are vulnerable to (among others) replay attacks, phishing attacks and cryptanalysis/dictionary attacks. Furthermore, proposals using asymmetric cryptography to secure authentication are unsuitable for low-profile mobile devices with limited battery power. Large-scale wireless authentication systems using asymmetric key cryptography may experience substantial overheads from cryptography costs. In contrast, the proposed authentication realisation must be able to provide secure authentication without compromising performance.

The following subsections propose a mechanism to overcome the shortcomings of existing authentication protocols. This mechanism utilises the proposed dynamic key cryptography scheme in section 4.1.3 to strengthen the security of authentication. Although dynamic key cryptography is a form of symmetric cryptography, it is able to

resist security attacks on authentication [NWL+10] such as replay attacks and cryptanalysis attacks. Because of the mechanism's simplicity and the low cost of the dynamic key scheme and the protocols, there is no trade off between performance and security.

A ticket-based authentication protocol is adapted from the authentication protocol proposed in the authentication architecture in chapter 3. The original idea of the protocol is derived from the Kerberos protocol of using tickets for authentication. Because of the efficiency of the authentication protocol, ticket-based authentication is suitable for services running on low-performance mobile devices. In order to save battery power and storage, services running on mobile devices prefer to move authorisation controls and authentication to other services with greater computation and storage resources. The authentication service AS thus conducts an authorisation control service (not included in this model) to validate permission before continuing authentication verification. Dynamic key cryptography is then employed in the ticket-based authentication protocol to overcome the security weakness evident under replay attacks and cryptanalysis attacks.

4.4.1 The Ticket-Based Authentication Protocol

The concept underlying the ticket-based authentication protocol is that AS issues a ticket for u to authenticate with s. The idea is derived from the Kerberos authentication method. However, instead of using a long-term authentication key, the ticket-based authentication protocol uses a dynamic key for authentication. The protocol is therefore able to resist replay attacks.

In the ticket-based authentication protocol, a user u sends an access request to AS for authentication to service s. In the request, u claims that it is a member of user group UG. In this authentication protocol, AS conducts an authorisation verification service for UG for accessing s. Once UG is confirmed as having permission to access

s as a member of service group SG, AS issues an authentication ticket for u. u then uses this ticket and an authentication token to authenticate to s. The authentication protocol has the five following phases:

i. In the first phase, u sends the request to AS claiming that it is a member of UG and wants to access service s. In addition to the claimed user group identity and the service identity s, the message contains a nonce (N_u) and a hash value of the nonce exclusive-OR operation with the group authentication key of UG ($N_u \oplus s \oplus K_{UG}^{auth}$).

ii. In phase 2, after receiving the message from phase 1, AS validates the hash value. After the validation, AS can trust* that the message from phase 1 has been created by a member of UG. In the next step, AS verifies the authorisation and looks up the service group SG among the service groups of s to ensure that UG has the authorisation to access SG. If service group SG exists so that UG can access it, AS creates and sends to u an authentication ticket containing the two initial keys (that is, EK and IK), the expiry time of the ticket (T_{expire}) and the group identity UG encrypted by the service group authentication key K_{SG}^{auth}. The ticket can only be comprehended by members of service group SG. Besides the authentication ticket, AS also sends materials to u to create dynamic keys including the two initial keys (EK and IK) encrypted by the user group authentication key K_{UG}^{auth}.

iii. In phase 3, after receiving the authentication ticket and the materials, u sends the authentication ticket, SG and a nonce ($N_u^{'}$) as an authentication request to u.

iv. In phase 4, s receives and extracts EK and IK from the authentication ticket. It generates m temporary keys ($TK_1, \ldots, TK_m$) and then creates a message to exchange them with u. The temporary key exchange message is encrypted by EK. From the dynamic keys generation material ($EK, IK, TK_1, \ldots TK_m$), s generates a sequence of dynamic keys ($DK_1, DK_2, \ldots, DK_n$) by using the dynamic

*AS cannot trust that message 1 is sent by a member of UG because adversaries can replay message 1.

key generation scheme. The temporary message key exchange is added $N_u^{'}$ to be a response to the message in phase 3. Later, s also sends another message containing $N_s^{'}$ to challenges u to encrypt $N_s^{'}$. This second message is encrypted by the first key (DK_1) in the dynamic key sequence.

v. In phase 5, once u receives the first message, it extracts $TK_1, \ldots, TK_m$ and generates the dynamic key sequence using the dynamic key generation scheme with EK, IK and $TK_1, \ldots, TK_m$. It uses the first dynamic key in the sequence (DK_1) to decrypt the second message from s and then validates $N_u^{'}$. Finally, u responds to the challenge of s by encrypting $N_s^{'}$ with the second dynamic key DK_2.

The protocol is formalised as follows:

1. $u \rightarrow AS : UG, N_u, s, h(N_u \oplus s \oplus K_{UG}^{auth})$.
2. $AS \rightarrow u : \{EK, IK, SG, N_u\}K_{UG}^{auth}, \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth}$.
3. $u \rightarrow s : SG, \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth}, N_u^{'}$.
4. $s \rightarrow u : \{TK_1, \ldots, TK_m\}EK, \{N_u', N_s^{'}\}DK_1$.
5. $u \rightarrow s : \{N_s^{'} + 1\}DK_2$.

The message flow in the five phases of the authentication protocol is described in figure 4.7.

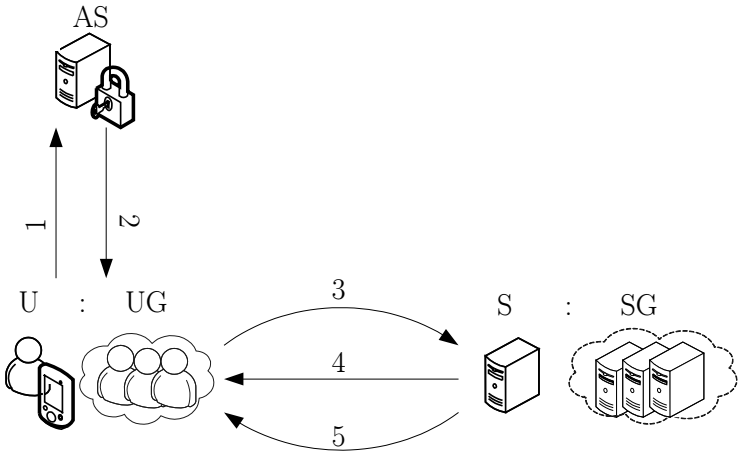


Figure 4.7: The Ticket-Based Authentication Protocol

At the end of a successful authentication, a secure channel between u and s is established. The channel is secured by the dynamic keys DK_3 and $DK_4 \ldots$ The material keys used to create the dynamic keys are created by AS and u and are exchanged in phases 2 and 3. The two keys K_{UG}^{auth} and K_{SG}^{auth} used in the key exchange are the group authentication keys of user group UG and service group SG respectively.

After a successful authentication between u and s, u can re-use the authentication ticket to authenticate to any services in the service group SG. In the ticket-based authentication protocol, phase 1 and phase 2 are used to obtain an authentication ticket from AS. Therefore, when u wants to re-authenticate, phase 1 and phase 2 are not repeated. While the ticket lifetime is still valid and the service group key K_{SG}^{auth} has not been rekeyed, the authentication ticket can be re-used for repeat authentication. The repeat authentication protocol thus has only three phases - phase 3, phase 4 and phase 5 - in the ticket-based authentication protocol.

4.4.2 The Request-Based Authentication Protocol

In the request-based authentication protocol, the service s receives a direct access request from the user u. In the request, u claims that it is a member of user group UG. First, s verifies whether UG is allowed to manage authorisation control when integrating with s. After confirming that members of UG can access s via permission from SG while s is a member of SG using the authorisation control service, s forwards the request to AS to ask for authentication. Finally, AS verifies the authenticity of the user group identity UG of user u and service group identity SG of service s.

The authentication protocol consists of six phases. The phases are described as follows:

i. In the first phase, u sends a request to s claiming that it is a member of UG. In addition to the claimed user group identity, the message contains a nonce (N_u)

and a hash value of the nonce exclusive-OR operation with the user group authentication key of UG ($N_u \oplus K_{UG}^{auth}$).

ii. In phase 2, after receiving message 1, s validates the authorisation of the request and then sends message 2 to AS. Message 2 contains information concerning the request in message 1 and also the claimed service group identity SG of s. In addition, s also adds into the message a nonce N_s and a hash value of the nonce exclusive-OR operation with the service group authentication key (K_{SG}^{auth})

iii. In phase 3, after receiving the message from s, AS generates the two keys EK and IK as material keys for s and u to generate dynamic keys. AS produces two messages and sends these to s. The first message is the material to generate dynamic keys for u. It contains the two keys EK and IK and the nonce N_u encrypted with the user group authentication key K_{UG}^{auth}. The second message includes EK, IK and N_s encrypted by K_{SG}^{auth}. These messages are sent to s as the material to generate the dynamic key for both s and u.

iv. In phase 4, s decrypts the second message from AS in phase 3 to extract EK and IK. s also forwards the first message received from AS (the key material for u) to u. A message is generated and sent to u containing m temporary keys $TK_1, \ldots, TK_m$ (as in step 2) to create dynamic keys. The message also includes another nonce ($N_s^{'}$) as a challenge for u to encrypt.

v. In phase 5, u extracts EK, IK and N_u from the key material message. The extracted value of N_u is compared with the original value of nonce N_u when it was created. When the values match, u can trust that EK and IK have been created by AS. u uses EK to decrypt the temporary keys $TK_1, \ldots, TK_m$ and the nonce $N_s^{'}$ from the second message. u uses the initial key IK and temporary keys $TK_1, \ldots, TK_m$ to create SK and then DK_1 (as in steps 3 and 4) to create a sequence of dynamic keys. Finally, u creates a new nonce ($N_u^{'}$) and combines a message with $N_u^{'}$ and $N_s^{'} + 1$ encrypted by DK_1 in response to the challenge of s.

vi. In the last phase, s also computes the dynamic keys DK_1 and DK_2 and uses DK_1 to extract the message from u in phase 5. s then validates $N_s^{'}$ to authenticate u. After the validation process is completed, s responds to u with the value $N_u^{'} - 1$ encrypted by DK_2 as confirmation of authentication and its willingness to serve u. After this phase, u and s continue the session and use the next dynamic keys ($DK_i, i > 2$) in the sequence as the cryptographic key to secure the communication.

The messages in the request-based authentication protocol are specified as follows:

1. $u \rightarrow\ s : UG, N_u, h(N_u \oplus K_{UG}^{auth})$
2. $s \rightarrow\ AS : UG, N_u, h(N_u \oplus K_{UG}^{auth}), SG, N_s, h(N_s \oplus K_{SG}^{auth})$
3. $AS \rightarrow\ s : \{EK, IK, N_u\}K_{UG}^{auth}, \{EK, IK, N_s\}K_{SG}^{auth}$
4. $s \rightarrow\ u : \{EK, IK, N_u\}K_{UG}^{auth}, \{TK_1, \ldots, TK_m, N_s'\}EK$
5. $u \rightarrow\ s : \{N_s' + 1, N_u'\}DK_1$
6. $s \rightarrow\ u : \{N_u' - 1\}DK_2$

The message flow in the six phases of the authentication protocol is described in figure 4.8.

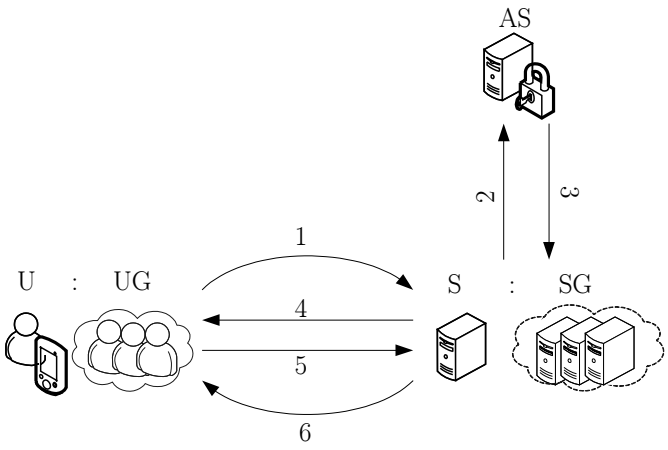


Figure 4.8: The Request-Based Authentication Protocol

Similar to the ticket-based authentication protocol, the result of the authentication protocol is the establishment of a secure channel between u and s. The secure channel uses a sequence of dynamic keys as cryptographic keys for communication. In phases 2, 3 and 4 during the authentication, s exchanges EK and IK indirectly via AS. In phase 4, s directly sends the temporary keys $TK_1, \ldots TK_m$ to u so that both u and s have enough material to generate a sequence of dynamic keys. In phase 5, u uses the dynamic key DK_1 to encrypt the authentication message and sends this to s. During phases 5 and 6, both u and s start to use the shared sequence of dynamic keys $DK_1, DK_2, \ldots$ in order to secure the communication between them.

Each authentication protocol has its own advantages and disadvantages. The computation cost of the authentication service of the request-based authentication protocol is less than that of the ticket-based authentication protocol. In contrast, the authentication cost of the service of the ticket-based authentication protocol is smaller than that of the request-based authentication protocol. While the ticket-based authentication protocol may create a bottleneck problem for the authentication service, on the other hand, efficient repeat authentication can only be performed by the ticket-based authentication protocol. These advantages and disadvantages are shown in table 4.3. The detailed performance analysis in the next chapter will further demonstrate these advantages and disadvantages.

Table 4.3: Comparison of Two Authentication Protocols

	The Ticket-Based Authentication Protocol	The Request-Based Auth Protocol
Authorisation	allows centralised authorisation control by AS	allows services to control authorisation themselves
Services running on dedicated servers	unsuitable	suitable
Services running on mobile devices	suitable	unsuitable
Number of messages	5	6

4.5 Summary

In this chapter, we demonstrated a realisation of the AMUS authentication model. Two main components were realised in the proposed authentication: the group manager and the authentication controller. The group manager controls user group memberships via group keys. In the authentication controller, users and services use their group keys to perform authentication via authentication protocols. These two components are realised based on two basic mechanisms: dynamic key cryptography and group key management.

Dynamic keys are one-time symmetric cryptographic keys that secure communication messages. In this chapter, we proposed a family of dynamic key generation schemes to generate dynamic keys. The family has two different schemes: one to generate the initial sequence of dynamic keys and the other to generate further dynamic key sequences once the previous sequence of dynamic keys is consumed. These dynamic keys are used to secure authentication and communications among entities and components in the model. The analysis in the next chapter will prove that the family of dynamic key generation schemes can enhance security and reduce the risks associated with cryptanalysis attack and replay attack. In addition to the dynamic key generation schemes, a membership-oriented group key management (MOGKM) scheme was developed to manage memberships of user and service groups.

The MOGKM structure is divided into three layers: the entity layer, the key-group layer and the detail layer. Each layer adapts to different requirements. The entity layer (the top level) contains the original user groups and/or service groups. The key-group layer contains key-groups for entities with multiple memberships. Key-groups in this layer are used to optimise rekeying operations relating to multiple memberships of individual members. The detail layer is a hybrid group key management structure adapted to wireless networks.

Based on this structure, MOGKM has four rekeying operations: member join, member leave, group switch and handoff. The rekeying for member join contains three steps: updating keys in the detail layer; updating keys in the key-group layer; and distributing keys to the entity. The rekeying for member leave also has three steps: de-registering the member; updating keys in the detail layer; and updating keys in the key-group layer. The group switch is a combination of a member leave and a member join operation. It contains three steps according to the rekeying requirements of the different layers. Finally, the member handoff operation has two steps: the first step is preparing the handoff and the second step is performing rekeying in the detail layer after the handoff.

MOGKM was extended with an extra rekeying operation in order to realise the group manager. Each entity group in the group manager has two keys: the group key from MOGKM and the authentication key. To reduce cryptanalysis attacks on the authentication keys of user groups and service groups, a periodic rekeying operation was proposed. Depending on the security requirements of each entity group, the group manager periodically invokes this rekeying operation to refresh authentication keys at regular intervals.

The authentication controller of the proposed authentication has two authentication protocols to ensure compatibility with authorisation control. Ticket-based authentication protocol is designed to suit services running on low-performance mobile devices. It has five authentication messages. In this protocol, users send requests to the authentication service to ask for authentication tickets. They can use the tickets to authenticate to services, similar to the Kerberos authentication protocol. The second authentication protocol is the request-based authentication protocol. This protocol is designed for services controlling authorisation themselves. It contains six authentication messages. In this protocol, users send the requests directly to the services, similar to authentication methods for web applications. While the ticket-based protocol is

best suited to services running on mobile devices, the request-based authentication protocol is best suited to services running on dedicate servers.

In the following chapter, chapter 5, both the proposed mechanisms and the authentication realisation from this chapter are discussed and analysed. The security and efficiency of the proposed authentication and its mechanism are examined and analysed in order to validate the realisation. The security and efficiency of the components of the authentication realisation, such as dynamic keys and group key management, are also investigated in detail. Based on the analysis, discussion and comparison of the components, the realisation and the proposed authentication model, we can then determine whether the AMUS model enables security, efficiency, scalability and flexibility for authentication in wireless networks.

Chapter 5

Analysis of the AMUS Authentication Realisation

In the previous chapter, an authentication realised from the AMUS authentication model was described. The two components (the group manager and the authentication controller) in the authentication rely on two mechanisms (dynamic key cryptography and MOGKM) to secure the authentication. MOGKM is used to realise the group manager while dynamic key cryptography is adopted in two authentication protocols of the authentication controller. The group manager is designed to manage group memberships and distribute group keys for the authentication controller. Security of the proposed authentication thus depends both on the components and the mechanisms employed to realise them. The two components are separately realised and perform independently.

In this chapter, the security and performance of the proposed authentication presented in chapter 4 are analysed and discussed. The mechanisms and realised components are examined. We then formally analyse the combination of these components; the realisation as a whole. Discussion is provided to support the analysis of the realisation in terms of the extra security features and the ability of the realisation to resist

attacks. The discussion also contains a comparison of the security and performance features of the proposed authentication from chapter 4 with existing authentication methods.

The analysis and discussion in this chapter have a number of purposes. The first purpose is to validate the correctness of the proposed authentication and its ability to provide strong authentication by examining the security of the proposed authentication realisation. The second purpose it to verify the efficiency of the proposed authentication by analysing and comparing the performance of the realisation to the other authentication methods described in chapter 2. Finally, the third purpose is to examine the ability of the proposed authentication to meet the four desired properties of security, efficiency, scalability and flexibility in authentication for wireless network users and services.

The structure of this chapter is organised as follows. In the first section, we present a formal security analysis of the mechanisms (MOGKM and the dynamic key cryptography) used in the authentication, the components (the group manager and the authentication controller) and the authentication as a whole. The second section analyses the performance of the components (the group manager and the authentication controller) and the authentication as a whole. The analysis results found in the first and the second sections are used in the third section to compare and discuss the security and efficiency features of the authentication realisation and its proposed mechanisms (that is, dynamic key cryptography and MOGKM) with the existing approaches originally described in chapter 2. This section also includes an extension of the AMUS authentication model to support audit tracing. The chapter concludes with a summary.

5.1 Security Analysis

The security analysis in this section is presented using a bottom-up approach for each mechanism and component of the entire authentication realisation. First, the security features of MOGKM and dynamic key cryptography are investigated. From the analysis of the dynamic key cryptography, the authentication protocols of the authentication controller are then analysed using a formal method. Later, the two authentication protocols are validated using security attacks to confirm their security. Finally, we present the proof of the security of the authentication based on the combination of the group manager and the authentication controller.

5.1.1 Security Analysis of Membership Oriented Group Key Management

The security of MOGKM reflects the security of the process of authentication key distribution among user groups and service groups. Because group keys in the group manager are also group authentication keys for user groups and service groups, the security of MOGKM is very important in this realisation. MOGKM warrants that only valid members can obtain group authentication keys. Authentication is thus able to prevent unauthorised access and phishing attacks. MOGKM therefore plays an important role in enhancing security for the authentication realisation.

MOGKM ensures the security requirements for group key management are met. The five security requirements (mentioned in previous chapters) are forward secrecy, backward secrecy, collusion resistance, key independence and minimal trust. Because MOGKM is customised and extended from [Wan09], it inherits security features from its predecessor. As in the proof in [Wan09], forward and backward secrecy are obtained from the rekeying operations of member join, member leave and member switch. Evidence of key independence, collusion resistance and minimal trust in rekeying operations is shown in [Wu09] and [Wan09]. These security features show

that the proposed MOGKM offers security when managing and distributing authentication keys to authorised members.

5.1.2 Security Analysis of Dynamic Key Cryptography

In this analysis, dynamic keys generated in section 4.1.3 are investigated to determine whether dynamic key cryptography is sufficiently secure to support our proposed authentication. The belief in the goodness of cryptographic keys is mentioned in [BAN90] as a base from which to construct the logic to verify authentication protocols. In addition to goodness, the freshness of authentication keys is also mentioned as the authentication key is not used prior to the current run of the authentication. The following theorem is used to explain the goodness and the freshness of dynamic keys from their dynamic key generation input:

Theorem 5.1. *If an entity P believes that two keys EK and IK are produced and sent by AS and it also believes in the freshness of either of the initial keys EK and IK or temporary keys $TK_1, \ldots, TK_m$, the produced dynamic keys $DK_1, \ldots, DK_n$ are believed to be good and fresh keys to communicate with other entities in an authentication.*

Proof. All entities in an authentication are assumed to believe that AS has jurisdiction over EK and IK. In other words, they believe that AS generates good keys EK and IK. Therefore, it is deduced that P believes in the goodness of EK and IK. Expressed another way:

$$\begin{aligned} &P \text{ believes } AS \text{ controls } EK, IK \wedge P \text{ believes } AS \text{ says } EK, IK \rightarrow \\ &P \text{ believes } EK, IK. \end{aligned} \tag{5.1}$$

With the corollary 4.1 and the collision freedom condition for the strong hash function $f(.)$, the goodness of the first dynamic key DK_1 (derived from equation (4.5) based on

EK and IK) can be deduced:

$$P \text{ believes } EK, IK \rightarrow P \text{ believes } DK_1. \tag{5.2}$$

From equation (4.5), when entity P believes in DK_1, P also believes in the next dynamic key DK_2 derived from equation (4.6). Therefore, the other dynamic keys in the sequence, $DK_3, DK_4, \ldots, DK_n$ are also believed by P.

The freshness of the first dynamic key DK_1 in the sequence is also deduced from the freshness of one of the initial keys (either EK, IK, or $TK_1, \ldots, TK_m$). Because DK_1 is computed by the collision free one-way hash function $f(.)$ with input parameters $TK_1, \ldots, TK_m$ and SK, the freshness of either $TK_1, \ldots, TK_m$ or SK can warrant the freshness of DK_1. However, SK is computed by $IK, TK_1, \ldots, TK_m$ in equation (4.4). Therefore, SK is fresh if one of the keys (IK or $TK_1, \ldots, TK_m$) is fresh. Hence the first dynamic key DK_1 is fresh.

Similar to calculating the goodness of other dynamic keys in the sequence, the dynamic keys $DK_2, DK_3, \ldots, DK_n$ are also fresh when one of the initial keys (IK, TK_1, $\ldots, TK_m$) is fresh. □

The theorem 5.1 is used to support the security analysis of the authentication protocols in the authentication controller in the next subsection.

5.1.3 Security Analysis of the Authentication Protocols

In this subsection, the request-based authentication protocol and the ticket-based authentication protocol described in section 4.4 are analysed by SVO logic to determine whether they achieve the six authentication goals. This formal method is chosen to analyse the two proposed authentication protocols because of its simplicity and the desired properties for authentication. There is no session key in either of the two

authentication protocols as it is replaced by the dynamic keys, $DK_i, \forall i > 0$, as cryptographic keys to secure communications between u and s. With the dynamic key DK_i, the six authentication goals of SVO in chapter 2 are rewritten as follows:

G1. u believes s says X

G2. u believes (s says $F(X, N_u^{'}), fresh(N_u^{'})$)

G3. u believes $u \overset{DK_i}{\longleftrightarrow} s$

G4. u believes $fresh(DK_i)$

G5. u believes s says ($u \overset{DK_i}{\longleftrightarrow} s$)

G6. u believes ($u \overset{DK_i}{\longleftrightarrow} s \wedge s$ says $\{N_u^{'}\}DK_i$)

These six authentication goals must be derived for both u and s in the two authentication protocols. G1 explains that P (either u or s) knows that Q is alive. G2 explains that Q says something relevant to the present conversation (via the nonce $N_P^{'}$). G3 explains that a secure channel with Q is established using dynamic keys DK_i. G4 explains that the dynamic keys securing the communication channel with Q are fresh. G5 explains that Q has a mutual understanding of the shared dynamic keys DK_i with P. Finally, G6 explains that Q confirms that it knows the current dynamic key by encrypting the nonce $N_P^{'}$.

Before analysing the request-based authentication protocol and the ticket-based authentication, we introduce two corollaries related to dynamic keys. The first corollary relates to services creating temporary keys $TK_1, \ldots, TK_m$; the second relates to users receiving $TK_1, \ldots, TK_m$. The corollaries interpret that when entities believe in received dynamic key materials EK, IK and $TK_1, \ldots, TK_m$, they also believe in their produced dynamic keys. In other words, both the goodness and the freshness of the dynamic keys DK_i are derived from the initial materials EK, IK, and $TK_1, \ldots, TK_m$. These two corollaries are used to prove that the two authentication protocols in sections 4.4.2 and 4.4.1 meet the mutual understanding of the dynamic key goal **G5** and the dynamic key confirmation goal **G6**.

Corollary 5.1. *First Dynamic Key Generation Corollary*

$$P \text{ believes } AS \text{ said } (EK, IK) \wedge P \text{ believes } fresh(TK_1, \ldots, TK_m) \wedge P \text{ says } (TK_1, \ \ldots, TK_m) \longrightarrow P \text{ believes } P \overset{DK_i}{\longleftrightarrow} Q \wedge P \text{ believes } fresh(DK_i), \forall i > 0 \tag{5.3}$$

The first dynamic key generation corollary explains that after receiving EK and IK from AS, P can create $TK_1, \ldots, TK_m$ and send them to Q to generate secure dynamic keys DK_i. P can believe in the goodness and the freshness of dynamic keys when communicating with Q.

Corollary 5.2. *Second Dynamic Key Generation Corollary*

$$Q \text{ believes } AS \text{ says } (EK, IK) \wedge Q \text{ believes } fresh(EK, IK) \wedge Q \text{ believes } P \text{ said } (TK_1, \ldots, TK_m) \longrightarrow Q \text{ believes } P \overset{DK_i}{\longleftrightarrow} Q \wedge Q \text{ believes } fresh(DK_i), \forall i > 0 \tag{5.4}$$

The second dynamic key generation corollary explains that after receiving and verifying the freshness of EK and IK from AS, Q can use $TK_1, \ldots, TK_m$ received from P to generate dynamic keys DK_i to communicate with P. Q can also believe in the goodness and the freshness of dynamic keys when communicating with P.

Both the above two corollaries are deduced from the theorem 5.1. The first dynamic key generation corollary is derived from the belief of P in both the freshness and the goodness of the temporary keys $TK_1, \ldots, TK_m$ generated by itself. The belief of Q in both the goodness and freshness of the pair EK and IK infers the second dynamic key generation corollary. With these two corollaries, the security of the ticket-based authentication protocol in section 4.4.1 and the request-based authentication protocol in section 4.4.2 are validated by SVO.

Security Analysis of the Ticket-Based Authentication Protocol

The concept underlying ticket-based authentication in section 4.4.1 is derived from Kerberos. A user u sends a request to AS in order to ask for a ticket to authenticate to a service s. AS validates u's authentication and authorisation of the request and then issues a ticket with a pair (EK and IK). The ticket also contains a timestamp (T_{expire}) to limit the lifetime of the ticket. u sends the received ticket to s asking to access s. s generates $TK_1, \ldots, TK_m$ and then sends it to u so that both u and s can compute a sequence of dynamic keys (DK_i). u and s can use dynamic keys DK_i to establish a secure communication channel between them. Before analysing the protocol, the following initial assumptions are made.

Initial State Assumptions

P1. u believes $u \overset{K_{UG}^{auth}}{\longleftrightarrow} AS$

P2. s believes $s \overset{K_{SG}^{auth}}{\longleftrightarrow} AS$

P3. u believes AS controls $u \overset{K}{\longleftrightarrow} s$

P4. s believes AS controls $u \overset{K}{\longleftrightarrow} s$

P5. u believes AS controls $fresh(EK, IK)$

P6. s believes $fresh(N_s^{'})$

P7. u believes $fresh(N_u)$

P8. u believes $fresh(N_u^{'})$

P9. s believes $TK_i, \forall i = 1 \ldots m$

P10. s believes $fresh(TK_i), \forall i = 1 \ldots m$

P1 and P2 state that both u and s are assumed to believe their group authentication keys. P3 and P4 state that they also believe in the two key EK and IK generated by AS. P5 explains that u believes in the freshness of the initial keys generated by AS. P6, P7 and P8 explain that u and s believe the nonces generated by themselves. P9 and P10 state that s believes in the freshness and the goodness of temporary keys $TK_1, \ldots, TK_m$ created by itself.

Received Message Assumptions

P11. AS received $(UG, N_u, s, h(N_u \oplus s \oplus K_{UG}^{auth}))$.

P13. u received $\{EK, IK, SG, N_u\}K_{UG}^{auth}, \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth}$.

P14. s received $(SG, \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth}, N_u^{'})$.

P15. u received $\{TK_1, \ldots, TK_m\}EK, \{N'_u, N'_s\}DK_1$.

P16. s received $\{N'_s + 1\}DK_2$.

Comprehension assumptions

P17. AS believes AS received $(UG, \langle N_u \rangle_{*AS}, \langle h(N_u \oplus s \oplus K_{UG}^{auth}) \rangle_{*AS})$.

P18. u believes u received $\{\langle EK, IK \rangle_{*u}, SG, N_u\}K_{UG}^{auth}, \langle \{UG, SG, EK, IK, T_{expire}\}$ $K_{SG}^{auth} \rangle_{*u}$.

P19. s believes s received $(SG, \{UG, SG, \langle EK, IK, T_{expire} \rangle_{*s}\}K_{SG}^{auth}, \langle N_u^{'} \rangle_{*s})$.

P20. u believes u received $\{\langle TK_1, \ldots, TK_m \rangle_{*u}\}\langle EK \rangle_{*u}, \{N'_u, \langle N'_s \rangle_{*u}\}\langle DK_1 \rangle_{*u}$.

P21. s believes s received $\{N_s^{'} + 1\}\langle DK_2 \rangle_{*s}$.

Interpretation assumptions

P22. u believes u received $\{\langle EK, IK \rangle_{*u}, SG, N_u\}K_{UG}^{auth}, \langle \{UG, SG, EK, IK, T_{expire}\}$ $K_{SG}^{auth} \rangle_{*u} \longrightarrow$ u believes u received $\{ u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s, \langle IK \rangle_{*u}, fresh(\langle EK, IK \rangle_{*s}),$ $SG, N_u\}K_{UG}^{auth}, \langle \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth} \rangle_{*u}$.

P23. s believes s received $(SG, \{UG, SG, \langle EK, IK, T_{expire} \rangle_{*s}\}K_{SG}^{auth}, \langle N_u^{'} \rangle_{*s}) \longrightarrow$ s believes s received $(SG, \{UG, SG, u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s, \langle IK, T_{expire} \rangle_{*s}\}K_{SG}^{auth}, \langle N_u^{'} \rangle_{*s})$.

P24. (u believes u received $\{\langle TK_1, \ldots, TK_m \rangle_{*u}\}\langle EK \rangle_{*u}, \{N'_u, \langle N'_s \rangle_{*u}\}\langle DK_1 \rangle_{*u}$) $\wedge$(u believes $u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s$) $\longrightarrow$ u believes u received $\{\langle TK_1, \ldots, TK_m \rangle_{*u},$ $u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s\}\langle EK \rangle_{*u}, \{N'_u, \langle N'_s \rangle_{*u}\}\langle DK_1 \rangle_{*u}$.

Derivations for u

i. u believes u received $\{ u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s, \langle IK \rangle_{*u}, fresh(\langle EK, IK \rangle_{*s}), SG, N_u\}K_{UG}^{auth},$ $\langle \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth} \rangle_{*u}$ by modus ponens using P22 and P18.

ii. u believes AS said $\{ u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s, \langle IK \rangle_{*u}, fresh(\langle EK, IK \rangle_{*s}), SG, N_u\}K_{UG}^{auth},$ $\langle \{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth} \rangle_{*u}$ by source association, (i), P1 and belief axioms.

iii. u believes AS says $\{ u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s, \langle IK \rangle_{*u}, fresh(\langle EK, IK \rangle_{*s}), SG, N_u\} K_{UG}^{auth}$, $\langle \{UG, SG, EK, IK, T_{expire}\} K_{SG}^{auth} \rangle_{*u}$ by freshness, none verification, (ii), P7 and belief axioms.

iv. u believes $u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s, \langle IK \rangle_{*u}$ by saying, jurisdiction, (iii), P3 and belief axioms.

v. u believes $fresh(\langle EK, IK \rangle_{*u})$ by saying, jurisdiction, (iii), P5 and belief axioms.

vi. u believes s said $\{\langle TK_1, \ldots, TK_m \rangle_{*u}, u \overset{\langle EK \rangle_{*u}}{\longleftrightarrow} s\} \langle EK \rangle_{*u}$ by source association, P24, (iv) and belief axioms.

vii. u believes $u \overset{DK_i}{\longleftrightarrow} s \wedge u$ believes $fresh(DK_i)$ by corollary 5.2, (iv), (v), (vi) and belief axiom.

viii. u believes s said $(N'_u, \langle N'_s \rangle_{*u}, u \overset{\langle DK1 \rangle_{*u}}{\longleftrightarrow} s)$ by source association, P24, (vii) and belief axioms.

ix. u believes s says $(N'_u, \langle N'_s \rangle_{*u}, u \overset{\langle DK1 \rangle_{*u}}{\longleftrightarrow} s)$ by freshness, nonce verification, (vii), (viii) and belief axioms.

From the analysis above, it can be seen that the six authentication goals (G1, ..., G6) in section 2.5.4 for u are met. For u, G1 is achieved in (ix); G2 in (ix) and P8; G3 and G4 in (vii); G5 and G6 in (ix). Similar to this, we conduct the derivations for s.

Derivations for s

i. s believes s received $(SG, \{UG, SG, u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s, \langle IK, T_{expire} \rangle_{*s}\} K_{SG}^{auth}, \langle N_u^{'} \rangle_{*s})$ by modus ponens using P23 and P19.

ii. s believes AS said $(SG, \{UG, SG, u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s, \langle IK, T_{expire} \rangle_{*s}\} K_{SG}^{auth}, \langle N_u^{'} \rangle_{*s})$ by source association, (i), P2 and belief axioms.

iii. s believes $u \overset{DK_i}{\longleftrightarrow} s \wedge s$ believes $fresh(DK_i)$ by corollary 5.1, (ii), P9, P10, P15 and belief axiom.

iv. s believes u said $(N_s^{'} + 1, u \overset{\langle DK_2 \rangle_{*s}}{\longleftrightarrow} s)$ by source association, P21, (iii) and belief axioms.

v. s believes u says $(N_s^{'}+1, u \overset{\langle DK_2 \rangle_{*s}}{\longleftrightarrow} s)$ by freshness, nonce verification, (iv) and belief axioms.

Similar to the derivations for u, the six authentication goals for s are met. For s, G1 is achieved in (v); G2 in (v) and P6; G3 and G4 in (iii); G5 and G6 in (v).

In summary, the ticket-based authentication protocol meets its authentication security goals. The derivations for u and s show that the six authentication goals are achieved. In the next section, the request-based authentication protocol in section 4.4.1 is analysed.

Security Analysis of the Request-Based Authentication Protocol

The request-based authentication protocol in section 4.4.2 has six messages. In the first message of the protocol, u sends an authentication request directly to s. However, u and s do not directly share any secret. Therefore, in the second message, s asks AS to authenticate u. To authenticate, u uses its group authentication key K_{UG}^{auth} while s utilises its group authentication key K_{SG}^{auth}. After authenticating with AS, u and s receive the pair EK and IK and use them to produce a sequence of dynamic keys (DK_i). The sequence of dynamic keys is then used to establish a secure channel between u and s. The following analysis validates that the request-based authentication protocol meets the six goals for authentication in SVO. Before starting the analysis, the following assumptions are made.

Initial State Assumptions

P1. u believes $u \overset{K_{UG}^{auth}}{\longleftrightarrow} AS$

P2. s believes $s \overset{K_{SG}^{auth}}{\longleftrightarrow} AS$

P3. u believes AS controls $u \overset{K}{\longleftrightarrow} s$

P4. s believes AS controls $u \overset{K}{\longleftrightarrow} s$

P5. u believes AS controls $fresh(EK, IK)$

P6. s believes AS controls $fresh(EK, IK)$

P7. u believes $fresh(N_u)$

P8. u believes $fresh(N_u^{'})$

P9. s believes $fresh(N_s)$

P10. s believes $fresh(N_s^{'})$

P11. s believes $TK_i, \forall i = 1 \ldots m$ P12. s believes $fresh(TK_i), \forall i = 1 \ldots m$

P1 and P2 state that both u and s are assumed to believe their group authentication keys. P3 and P4 explain that both u and s also believe in the two keys EK and IK generated by AS. P5 and P6 explain that they believe in the freshness of the keys generated by AS. P7, P8, P9 and P10 state that u and s believe the nonces generated by themselves. P11 and P12 explain that s believes in the freshness of temporary keys $TK_1, \ldots, TK_m$ created by itself.

Received Message Assumptions

P13. s received $(UG, N_u, h(N_u \oplus K_{UG}^{auth}))$.

P14. AS received $(UG, N_u, h(N_u \oplus K_{UG}^{auth}), SG, N_s, h(N_s \oplus K_{SG}^{auth}))$.

P15. s received $\{EK, IK, N_u\}K_{UG}^{auth}, \{EK, IK, N_s\}K_{SG}^{auth}$.

P16. u received $\{EK, IK, N_u\}K_{UG}^{auth}, \{TK_1, \ldots, TK_m, N_s'\}EK$.

P17. s received $\{N_s' + 1, N_u'\}DK_1$.

P18. u received $\{N_u' - 1\}DK_2$.

Comprehension assumptions

P19. s believes s received $(UG, \langle N_u \rangle_{*s}, \langle h(N_u \oplus K_{UG}^{auth}) \rangle_{*s})$.

P20. AS believes AS received $(UG, \langle N_u \rangle_{*AS}, h(\langle N_s \rangle_{*AS} \oplus K_{UG}^{auth}), SG, \langle N_s \rangle_{*AS}, h(\langle N_s \rangle_{*AS} \oplus K_{SG}^{auth}))$.

P21. s believes s received $\langle \{EK, IK, N_u\}K_{UG}^{auth} \rangle_{*s}, \{\langle EK, IK \rangle_{*s}, N_s\}K_{SG}^{auth}$.

P22. u believes u received $\{\langle EK, IK \rangle_{*u}, N_u\}K_{UG}^{auth}, \{TK_1, \ldots, TK_m, \langle N_s' \rangle_{*u}\}\langle EK \rangle_{*u}$.

P23. s believes s received $\{N_s' + 1, \langle N_u' \rangle_{*s}\}\langle DK_1 \rangle_{*s}$.

P24. u believes u received $\{N_u' - 1\}\langle DK_2 \rangle_{*u}$.

Interpretation assumptions

P25. s believes s received $\langle \{EK, IK, N_u\}K_{UG}^{auth} \rangle_{*s}, \{\langle EK, IK \rangle_{*s}, N_s\}K_{SG}^{auth}$

$\longrightarrow$ s believes s received $\langle \{EK, IK, N_u\}K_{UG}^{auth} \rangle_{*s}, \{N_s, \langle IK \rangle_{*s}, u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s$,

$fresh(\langle EK, IK\rangle_{*s}\}K_{SG}^{auth}$.

P26. u believes u received $\{\langle EK, IK\rangle_{*u}, N_u\}K_{UG}^{auth}, \{TK_1, \ldots, TK_m, \langle N_s^{'}\rangle_{*u}\}\langle EK\rangle_{*u}$ $\longrightarrow$ u believes u received $\{u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \langle IK\rangle_{*u}, N_u, fresh(\langle EK, IK\rangle_{*s})\}K_{UG}^{auth}$, $\{(TK_1, \ldots, TK_m, \langle N_s^{'}\rangle, u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \}\langle EK\rangle_{*u})$.

P27. s believes s received $\{N_s^{'} + 1, \langle N_u^{'}\rangle_{*s}\}\langle DK_1\rangle_{*s}$ $\longrightarrow$ s believes s received $\{N_s^{'}, \langle N_u^{'}\rangle_{*u}, u \overset{\langle DK_1\rangle_{*s}}{\longleftrightarrow} s\}\langle DK_1\rangle_{*u}$.

P28. u believes u received $\{N_u^{'} - 1\}\langle DK_2\rangle_{*u}$ $\longrightarrow$ u believes u received $\{N_u^{'}$, $u \overset{\langle DK_2\rangle_{*s}}{\longleftrightarrow} s\}\langle DK_2\rangle_{*u}$.

Derivations for u

i. u believes u received $\{u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \langle IK\rangle_{*u}, N_u, fresh(\langle EK, IK\rangle_{*s})\}K_{UG}^{auth}$, $\{(TK_1, \ldots, TK_m, \langle N_s^{'}\rangle, u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \}\langle EK\rangle_{*u})$ by modus ponens using P26 and P22.

ii. u believes AS said $\{u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \langle IK\rangle_{*u}, N_u, fresh(\langle EK, IK\rangle_{*s})\}K_{UG}^{auth}$ by source association, (i), P1 and belief axioms.

iii. u believes AS says $\{u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \langle IK\rangle_{*u}, N_u, fresh(\langle EK, IK\rangle_{*s})\}K_{UG}^{auth}$ by freshness, none verification, (ii), P7 and belief axioms..

iv. u believes $u \overset{\langle EK\rangle_{*u}}{\longleftrightarrow} s, \langle IK\rangle_{*u}$ by saying, jurisdiction, (iii), P3 and belief axioms.

v. u believes $fresh(\langle EK, IK\rangle_{*u})$ by freshness axiom, (iii), P3 and P5.

vi. u believes s said $\{TK_1, \ldots, TK_m, \langle N_s^{'}\rangle, u \overset{\langle EK\rangle_{*s}}{\longleftrightarrow} s, \}\langle EK\rangle_{*u}$ by source association, (i), (iv) and belief axioms.

vii. u believes $u \overset{DK_i}{\longleftrightarrow} s \wedge u$ believes $fresh(DK_i)$ by corollary 5.2, (iv), (v) and (vi) and belief axiom.

viii. u believes s said $(N_u^{'}, u \overset{\langle DK_2\rangle_{*u}}{\longleftrightarrow} s)$ by source association P28, and (vii) and belief axioms.

ix. u believes s says $(N_u^{'}, u \overset{\langle DK_2\rangle_{*u}}{\longleftrightarrow} s)$ by freshness, nonce verification, (vii), (viii) and belief axioms.

From the analysis above, the six authentication goals for u are met. For u, G1 is achieved in (ix); G2 in (ix) and P8; G3 and G4 in (vii); G5 and G6 in (ix). Similar to u, we conduct the derivations for s.

Derivations for s

i. s believes s received $\{N_s, \langle IK \rangle_{*s}, u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s, fresh(\langle EK, IK \rangle_{*s}\}K_{SG}^{auth}$ by modus ponens using P25 and P21.

ii. s believes AS said $\{N_s, \langle IK \rangle_{*s}, u \overset{\langle EK \rangle_{*s}}{\longleftrightarrow} s, fresh(\langle EK, IK \rangle_{*s}\}K_{SG}^{auth}$ by source association, (i), P2 and belief axioms.

iii. s believes $u \overset{DK_i}{\longleftrightarrow} s \wedge s$ believes $fresh(DK_i)$ by corollary 5.1, (ii), P11, P12, P16 and belief axiom.

iv. s believes u said $(N_s^{'}, \langle N_u^{'} \rangle_{*u}, u \overset{\langle DK_1 \rangle_{*s}}{\longleftrightarrow} s)$ by source association, P27, (iii) and belief axioms.

v. s believes u says $(N_s^{'}, \langle N_u^{'} \rangle_{*u}, u \overset{\langle DK_1 \rangle_{*s}}{\longleftrightarrow} s)$ by freshness, nonce verification, (iv) and belief axioms.

Similar to the derivations for u, the derivations of s infer that the authentication for s satisfies the six authentication goals. For s, G1 is achieved in (v); G2 in (v) and P10; G3 and G4 in (iii); G5 and G6 in (v).

Similar to the ticket-based authentication protocol, the request-based authentication protocol also meets the six goals. In the following section, these two proposed authentication protocols of the authentication controller are validated using three well-known attack methods for authentication protocols.

5.1.4 Validation of the Authentication Controller using Possible Attacks

In this section, the authentication controller and two proposed authentication protocols are validated using common attack methods described in section 2.4. The validation is used to confirm the security analysis for the authentication protocols and the

authentication controller. In section 2.4, three common types of security attacks on authentication protocols were described: replay attacks, phishing attacks and cryptanalysis attacks. The following theorem 5.2, property 5.1 and property 5.2 show that the proposed authentication protocols are secure under these three types of security attacks.

Theorem 5.2. *The request-based authentication protocol in section 4.4.2 and the ticket-based authentication in section 4.4.1 are secure under replay attacks.*

To prove the theorem, two different types of replay attacks (described in section 2.4.1) on the request-based authentication protocol and the ticket-based authentication protocol are examined.

External Replay Attacks

In external replay attacks, the symmetric cryptography using shared authentication keys (K_{UG}^{auth} and K_{SG}^{auth}) and the keys themselves are assumed to be infeasible to break by cryptanalysis. The assumption is reasonable because they are rekeyed by the periodic rekeying operation in section 4.3.1. However, the session keys, temporary keys and dynamic keys ($EK, IK, TK_1, \ldots, TK_m, DK_1$ and DK_2) are breakable by brute force if given a sufficient amount of time. The following analysis finds the possibility of a replay attack using a dynamic key from a past authentication.

The Ticket-Based Authentication Protocol

Among the five messages in the ticket-based authentication protocol in section 4.4.1, there are three items named I1, I2 and I3:

I1. $\{EK, IK, SG, N_u\}K_{UG}^{auth}$

I2. $\{UG, SG, EK, IK, T_{expire}\}K_{SG}^{auth}$

I3. $\{TK_1, TK_2\}EK$

These items are the ticket and tokens used in distributing the initial keys and the temporary keys to create dynamic keys (no session key is distributed). I1 can be found in message 2. I2 can be found in messages 2 and 3. I3 is in message 4. (Note - ticket I1 can be re-used in future authentications.) These three items are the main targets for replay attacks.

Lemma 5.1. *The ticket-based authentication protocol is secure under external replay attacks.*

Proof. Assume that two dynamic keys (DK_1 and DK_2) from a previous authentication are compromised. To obtain unauthorised access by replaying the three items I1, I2 and I3, an adversary must be able to persuade services to regenerate exactly the same dynamic keys; that is, DK_1 and DK_2. In the ticket-based authentication protocol, the service is not able to verify the freshness of EK and IK generated by AS. Therefore, an adversary can replay the authentication ticket containing the pair EK and IK. (Note - the control and generation of the temporary keys $TK_1, \ldots, TK_m$ are managed by the service s.) The probability of the temporary keys, $TK_1^{'}, \ldots, TK_m^{'}$, being generated with exactly the same values as the previous authentication ($TK_1, \ldots, TK_m$) is less than or equal to $\frac{1}{2^{2s}}$ where s is a bit length of the temporary keys. Hence, the probability of convincing the service to generate the same DK_1 and DK_2 to re-use I1, I2 and I3 for authentication is less than or equal to $\frac{1}{2^{2s}}$. In other words, it is infeasible to obtain unauthorised keys using replay attacks in the ticket-based authentication protocol. In summary, the ticket-based authentication protocol is able to detect and resist external replay attacks. □

The Request-Based Authentication Protocol

Among the six messages in the request-based authentication protocol in section 4.4.2, there are also three items named as I1, I2 and I3.

I1. $\{EK, IK, N_u\}K_{UG}^{auth}$

I2. $\{EK, IK, N_s\}K_{SG}^{auth}$

I3. $\{TK_1, \ldots, TK_m, N'_s\}EK$

These three items are tokens used to distribute the initial keys EK and IK and the temporary keys $TK_1, \ldots, TK_m$ to create dynamic keys. I1 can be found in messages 3 and 4; I2 can be found in message 3; and I3 in message 4. None of these items are re-usable. These three items are also targets for replay attacks.

Lemma 5.2. *The request-based authentication protocol is secure under external replay attacks.*

Proof. The proof is similar to that of the ticket-based authentication protocol. Assume that DK_1 and DK_2 are two previously used and compromised authentication keys. To obtain unauthorised access by replaying the three items I1, I2 and I3, an adversary must be able to persuade services to regenerate exactly the same dynamic keys DK_1 and DK_2. To ask the service s to generate DK_1 and DK_2 exactly as in the previous authentication, the adversary needs to convince services that the initial keys and the temporary keys (that is, $EK, IK, TK_1, \ldots, TK_m$) are identical in their values to the previous authentication. In order to do that, the adversary interferes with communication between s and AS in message 3 to modify I1 and I2. However, the service s is also able to verify the freshness of the nonce N_s to detect the modification in this message. Furthermore, $TK_1, \ldots, TK_m$ are generated and controlled by service s. Under the request-based authentication protocol, an external replay attack that masquerades as a legal user can therefore be detected.

In another case, imagine that a hostile service masquerades as a legitimate service s to serve the requests of users. Similar to the process of masquerading as a legal user, the hostile service must be able to convince legal users to re-use the initial and temporary keys (that is, EK, IK and $TK_1, \ldots, TK_m$) from a previous authentication. The hostile service can replay messages having items I1 and I3 in message 4. However, users are also able to verify the nonce N_u in I1 to detect replay message 4 from the

masquerading service. In summary, we can conclude that the request-based authentication protocol is able to detect and resist external replay attacks. □

Internal Replay Attacks

Lemma 5.3. *Internal replay attacks cannot be mounted on the request-based and ticket-based authentication protocols.*

Proof. Internal replay refers to attacks using part of a previous message from the authentication protocols. In both the six phases of the request-based authentication protocol and the five phases of the ticket-based authentication, a cryptographic key is never repeated. Hence, it is infeasible for an internal penetrator to find a similar cryptographic item to replay a message in order to obtain unauthorised access. It is reasonable to conclude that neither of the two authentication protocols is vulnerable under internal replay attack. □

Proof. For theorem 5.2. From lemmas 5.1, 5.2 and 5.3, we can conclude that both the proposed authentication protocols are able to detect and resist known replay attacks. □

Property 5.1. *The request-based authentication protocol and the ticket-based authentication protocol are secure under cryptanalysis attacks.*

Cryptanalysis attacks (including dictionary attacks) are efforts to break cryptography based upon previously used captured messages. In authentication, cryptanalysis attacks target low entropy password/authentication keys used to encrypt messages in authentication protocols. By detecting patterns from captured ciphertexts and plaintext messages, an adversary may be able to correctly guess the cryptography key. In [BK98], Biryukov and Kushilevitz show that when an adequate amount of ciphertexts are captured, an adversary may be able to perform a successful cryptanalysis to deduce corresponding plaintext or even the cryptographic key. In other words,

a long-term re-usable authentication key that encrypts multiple messages over a long time period for many authentication attempts creates vulnerability to cryptanalysis attacks.

Neither of the two proposed authentication protocols uses session keys nor long-term permanent authentication keys to encrypt messages. In the two authentication protocols, session keys are replaced by dynamic keys that are one-time cryptographic symmetric keys. Even if an adversary were to gain access to a compromised cryptographic dynamic key, the adversary would be unable to obtain further unauthorised access. In addition, authentication keys are refreshed after a certain amount of time by periodic rekeying. Authentication keys are therefore replaced by new keys before they become vulnerable under cryptanalysis as determined by the security requirement for their groups. In summary, the realisation of the authentication controller has minimal risk under cryptanalysis attacks.

Property 5.2. *The request-based authentication protocol and the ticket-based authentication protocol are secure under phishing attacks.*

Phishing attacks are made on authentications lacking mutual authentication. This type of attack is likely to be successful where the user or service is able to provide the requested data without having to check with a third party that the request is legitimate. In both the request-based and ticket-based authentication protocols, users have mutual authentication with both the authentication service AS and the services. No authentication key is transferred directly between users and services or the authentication services. The involvement of trusted third parties ensures that phishing attacks cannot be performed successfully in the proposed realisation.

In the above sections, the security of MOGKM and the authentication protocols were verified. The verification showed that MOGKM provides effective group key management to secure group communication. The verification also showed that the

two authentication protocols, when used with dynamic keys, can resist attack to provide secure authentication in the authentication controller. In the next section, we examine the security of the combined realised group manager and the authentication controller.

5.1.5 Security Analysis of the Realisation of the AMUS Authentication Model

In the realisation, the authentication keys are shared between the authentication controller and the group manager. Each user group or service group has an authentication key corresponding to its group identity. In the group manager, the authentication keys are secured by the rekeying operations in MOGKM. In the authentication controller, the authentication keys are vital for authenticating user group and service group identities. Analysis presented in sections 5.1.3 and 5.1.4 proves that the authentication controller grants access only to valid entities having authorised authentication keys via one of the two proposed authentication protocols. The formal analysis in section 5.1.3 shows that the two authentication protocols are secure if the key secrecy and the forward key secrecy properties of the authentication keys are ensured. The conditions are presented in the assumptions P1 and P2 (the goodness of K_{UG}^{auth} and K_{SG}^{auth})) in section 5.1.3).

According to Merkle [Mer82], key secrecy [BP05] [BM03] and forward secrecy [ML08] for authentication keys are essential security requirements in authentication. Key secrecy explains that it is infeasible to compute the current cryptographic keys from any other random cryptographic keys. Forward key secrecy in authentication is achieved when a compromised cryptographic key from a previous authentication does not allow a passive adversary to deduce a current or subsequent authentication key in order to obtain unauthorised access in the future. The two following properties of the authentication realisation are used to confirm that the authentication keys in the AMUS model achieve key secrecy and forward key secrecy.

Property 5.3. *Authentication keys have key secrecy.*

In the group manager, the authentication keys are securely transferred in group communications by group keys in MOGKM. It is infeasible to compute authentication keys without having the group keys. Thus, because of the key secrecy property for MOGKM in [Wan09], the authentication keys (that is, K_{UG}^{auth} and K_{SG}^{auth}) also have the property of key secrecy.

In the authentication controller, the authentication keys K_{UG}^{auth} and K_{SG}^{auth} are used to secure the transferring initial keys (EK, IK) and the material to create dynamic keys. The authentication keys also validate the authenticity of AS via nonces N_u and N_s. No relationship exists between the authentication keys (K_{UG}^{auth} and K_{SG}^{auth}) and the other keys in the authentication controller. Thus it is infeasible to deduce the authentication keys from compromised past keys.

Property 5.4. *Authentication keys have forward key secrecy.*

The authentication keys are generated randomly and independently from former keys. In the periodic rekeying operation in section 4.3.1, the new values of the authentication keys are derived randomly from the key generation engine of the GKS. The new values thus have no connection to the old values. The three other rekeying operations, member join, member leave and member switch, also renew the authentication keys in the same way. It is thus infeasible to deduce the current authentication keys from the values of former keys.

Based on the above theorems and property, the following theorem proves the security of the authentication realisation of the AMUS model.

Theorem 5.3. *The realisation provides secure authentication for members of user groups and service groups.*

Proof. The property 5.3 and the property 5.4 explain that only authorised users and services can obtain group keys. According to the security analysis for the realisation

of the authentication controller in section 5.1.3, the proposed authentication protocols are secure in verifying authentication for user and service groups. In summary, the realisation warrants that only authorised users and services can perform authentication successfully and consequently access authorised services. Therefore, we can conclude that the authentication realisation is secure. □

The theorem 5.3 explains the security of the proposed authentication. It shows that authentication and authorisation to access valid services are only given to authorised users via their user groups and service groups. The authentication is able to resist common authentication attacks such as replay attack, cryptanalysis attack and phishing attack. The result of security analysis for the authentication realisation is used to compare with existing approaches in section 5.3.1.

5.2 Performance Analysis

In the previous section, we analysed the realisation of the AMUS authentication model and demonstrated the security of the authentication. We showed that the realisation proposed in chapter 4 is able to resist security attacks on authentication such as replay attacks, phishing attacks and cryptanalysis/dictionary attacks. Usually, to achieve high levels of security, authentication methods have to concede performance. To prove that the AMUS authentication realisation achieves both security and performance, in this section we analyse the performance of the authentication realisation.

The analysis utilises a bottom-up approach. We begin by analysing the performance of MOGKM and the group manager. We then conduct an analysis of the performance of the authentication protocols and the authentication controller. Because of the independence of the group manager and the authentication controller, the performance of these two components is unrelated. The performance of the authentication realisation is instead derived from the authentication protocols in the authentication controller.

The group manager and the authentication controller are independent because they have different functions. The group manager controls group membership management tasks including operations related to memberships of user groups and service groups. In addition to the periodic rekeying operation in section 4.3.1, the other rekeying operations are group membership management functions. These operations are not related to the authentication functions conducted by the authentication controller. Furthermore, the periodic rekeying operation is also independent of the authentication functions of the authentication controller. The key management layer of the group manager is transparent and independent of the authentication layer of the authentication controller. The performances of the two components are thus unconnected.

In this section, the cost of the group manager and the cost of the authentication controller are separately analysed. The cost of the group manager includes the costs of the rekeying operations in MOGKM. These costs do not contribute to the cost of the authentication functions in the authentication controller. The cost of both two types of operations are analysed based on the computation and communication costs. We begin the analysis by examining the performance of the membership management operations in the group manager and conclude the analysis by examining the performance of the authentication operations in the authentication controller.

5.2.1 Performance Analysis of the Membership Management Operations

In this section, the performance of the group membership management operations in the group manager is analysed through the performance of the rekeying operations in MOGKM. In the group manager, two types of groups (user and service) form two separate hierarchical tree structures for group membership management. Each hierarchical tree structure is independently managed by one sub-component of the group manager. The user group sub-component is managed by the UGM and the service

group sub-component is managed by the SGM. Both of these adopt MOGKM sharing a GKS and $z+1$ CKSs for handling group membership changes. Although the number of groups on the two trees is different, the trees have similar structures and are managed independently. They can thus be analysed as a single tree in MOGKM with a hierarchical structure but having different sizes. Without losing generality, and in order to simplify presentation, the key tree in MOGKM is assumed to be a completely balanced tree as in other group key management analyses ([LYGL01] [RH03] [PDM03]).

Notation mentioned in table 4.2.2 is repeated to help analyse the performance of MOGKM. Let the average height of the leader-cluster be h_l. The height of a member-cluster is h_m and the fixed degree of the trees is d. Let r be the number of memberships of the entities e joining or leaving. Let w be the number of indirectly affected key-groups from the leaving entity e. We use the analysis based on this notation in order to calculate the average costs of operations of MOGKM. Finally, a simulation is performed to verify the analysis.

A. Performance Analysis of Rekeying Operations in MOGKM

Rekeying for Member Join

When joining a group, the new entity e is assigned, in the detail layer, to one of two places: either the leader-cluster or a member-cluster.

Based on the location of e after joining, the cost of the rekeying operation for member join is calculated over three steps:

1. In step 1, rekeying in the detail layer, the number of encryptions depends on the location:

   (a) If e is going to be assigned to the leader-cluster, CKS generates h_l+1 group keys in the leader-cluster and a new key-group-key K_{KG_e}. The number

of encryptions in this situation is $h_l + 1$. CKS also sends one multi-cast message containing $h_l + 1$ keys to update the group keys.

(b) If e is going to be assigned to a member-cluster, CKS encrypts $h_m + 1$ keys and sends one multi-cast message containing $h_m + 1$ keys.

2. In step 2, rekeying for affected keys, GKS generates r new entity-group-keys corresponding to r memberships of the entity.

3. In the last step, GKS sends the new group keys to e. As in the first step, there are two cases:

(a) If e is assigned to the leader-cluster, GKS encrypts one message containing $h_l + r + 1$ keys and sends one uni-cast message to e containing $h_l + r + 1$ keys.

(b) If e is assigned to a member-cluster, GKS also encrypts one message containing $h_m + r + 1$ keys and sends one uni-cast message containing $h_m + r + 1$ keys.

Table 5.1 summarises the computation and communication costs for each of the steps in case (a).

Table 5.1: Rekeying Costs for the Member Join Operation in the Leader-Cluster

	Encryption Cost			Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	2	1	$h_l + 1$	1 uni-cast (r+4 keys)+1 uni-cast (1 key)	1 multi-cast (1 key) + 1 uni-cast (2 keys)	1 multi-cast ($h_l + 1$ keys)
Step 2		r			1 multi-cast (r keys)	
Step 3		$h_l + r + 1$			1 uni-cast ($h_l + r + 1$ keys)	

Table 5.2 summarises the computation and communication costs for each of the steps in case (b).

Table 5.2: Rekeying Costs for the Member Join Operation in a Member-Cluster

	Encryption Cost			Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	2	1	$h_m + 1$	1 uni-cast (r+4 keys) +1 uni-cast (1 key)	1 multi-cast (1 key) + 1 uni-cast (2 keys)	1 multi-cast ($h_m + 1$ keys)
Step 2		r			1 multi-cast (r keys)	
Step 3		$h_m + r + 1$			1 uni-cast ($h_m + r + 1$ keys)	

Rekeying for Member Leave

Similar to the member join operation, the cost of the rekeying for member leave is determined from three steps.

1. In step 1, GKS receives and verifies the leave request.

2. In step 2, the rekeying in the detail layer for member leave has four different situations depending on the original location of leaving entity e.

   (a) If e is from a member-cluster, CKS generates $h_m + 1$ new group keys in the member-cluster and one new K_{KG_e}. The updating process requires ($d \times h_m + 2$) encryptions. It also uses d uni-cast messages (1 key each), $h_m - 1$ multi-cast messages (d keys each) and two multi-cast messages (1 key).

   (b) If e is a leader-candidate from the leader-cluster, CKS generates h_l group keys in the leader-cluster and a new key-group-key K_{KG_e}. The number of encryptions in this case is $d \times h_l + z$. CKS sends ($d - 1$) uni-cast messages (1 key each), ($h_l - 1$) multi-cast messages (d keys each), and $z + 1$ multi-cast messages (1 key each) to update the group keys.

   (c) If e is a cluster-leader and a leader-candidate is available, CKS generates h_l group keys in the leader-cluster and a new key-group-key K_{KG_e}. The

number of encryptions in this case is $d \times (h_m + 1) + z + 1$. CKS sends (d) uni-cast messages (1 key each), (h_l) multi-cast messages (d keys each), and $z + 1$ multi-cast messages (1 key each) to update the group keys.

(d) If the leaving entity is a cluster-leader and no leader-candidate is available, CKS encrypts $d \times (h_l + h_m) + h_l + z$ keys. It sends $2d - 2$ uni-cast messages (1 key each), $(h_m + h_l - 2)$ multi-cast messages (d keys each), one uni-cast message ($h_l + 1$ keys) and $z + 1$ multi-cast messages (1 key each) to update the group keys.

3. In step 3, when rekeying the affected keys, GKS generates r new keys corresponding to r memberships of the entity. This step takes $r \times w$ encryptions and r multi-cast messages (w keys each).

The computation and communication costs for the rekeying of member leave in the four different cases are summarised in table 5.3, table 5.4, table 5.5 and table 5.6 respectively.

Table 5.3: Rekeying Costs for a Member Leaving a Member-Cluster

Encryption Cost			Communication Cost	
	GKS	CKS	GKS	CKS
Step 1	1		1 multi-cast (2 keys)	
Step 2		$d \times h_m + 2$		d uni-cast (1 key) + (h_m1) multi-cast (d keys) + 2 multi-cast (1 key)
Step 3	$r \times w$			r multi-cast (w keys)

Table 5.4: Rekeying Costs for a Leader-Candidate Leaving a Leader-Cluster

Encryption Cost			Communication Cost	
	GKS	CKS	GKS	CKS
Step 1	1		1 multi-cast (2 keys)	
Step 2		$d \times h_l + z$		d uni-cast (1 key) + (h_m - 1) multi-cast (d keys) + (z+1) multi-cast (1 key)
Step 3	$r \times w$			r multi-cast (w keys)

Table 5.5: Rekeying Costs for a Cluster-Leader Leaving with a Leader-Candidate

Encryption Cost			Communication Cost	
	GKS	CKS	GKS	CKS
Step 1	1		1 multi-cast (2 keys)	
Step 2		$d \times (h_l + 1) + z + 1$		d uni-cast (1 key) + h_l multi-cast (d keys) +(z+1) multi-cast (1 key)
Step 3	$r \times w$			r multi-cast (w keys)

Table 5.6: Rekeying Costs for a Cluster-Leader Leaving Lacking a Leader-Candidate

Encryption Cost			Communication Cost	
	GKS	CKS	GKS	CKS
Step 1	1		1 multi-cast (2 keys)	
Step 2		$d \times (h_l + h_m) + h_l + z$		$(2d - 2)$ uni-cast (1 key)+1 uni-cast ($h_l + 1$ keys)+($h_m + h_l - 2$) multi-cast (d keys) +(z+1) multi-cast (1 key)
Step 3	$r \times w$			r multi-cast (w keys)

Rekeying for Key-Group Switch

A group switch operation is a combination of a member leave operation and a member join operation. Because there are two cases of member join and four cases of member leave, the rekeying for a member switch operation involves eight different cases. The performance analysis of rekeying for member switch is based on these eight different cases.

(a) e is originally in a member-cluster of a key-group. e then switches to another member-cluster of another key-group;

(b) e is originally in a member-cluster of a key-group. e then switches to a leader-cluster of another key-group (and becomes a leader-candidate);

(c) e is originally a leader-candidate of a key-group. e then switches to a member-cluster of another key-group;

(d) e is originally a leader-candidate of a key-group. e then switches to another leader-cluster of another key-group (and becomes a leader-candidate);

(e) e is originally a cluster-leader of a key-group (with an available leader-candidate). e then switches to a member-cluster of another key-group;

(f) e is originally a cluster-leader of a key-group (with (an) available leader-candidate(s)). e then switches to another leader-cluster of another key-group (and becomes a leader-candidate);

(g) e is originally a cluster-leader of a key-group (with no available leader-candidate). e then switches to a member-cluster of another key-group; or

(h) e is originally a cluster-leader of a key-group (with no available leader-candidate). e then switches to another leader-cluster of another key-group (and becomes a leader-candidate).

The computation and communication costs of rekeying for the member switch operation in the eight different cases are summarised in tables 5.7, 5.8, 5.9, 5.10, 5.11, 5.12, 5.13 and 5.14, respectively.

Table 5.7: Rekeying Costs for the Member Switch Operation in Case (a)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_m + dh_m + 3$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	d uni-cast (1 key)+(h_m-1) multi-cast (d keys) + 2 multi-cast (1 key) + 1 multi-cast (h_m+1 keys)
Step 2		$rw + s$			1 multi-cast (s keys) + r multi-cast (w keys)	
Step 3		$h_m + q + 1$			1 uni-cast ($h_m + q + 1$ keys)	

Table 5.8: Rekeying Costs for the Member Switch Operation in Case (b)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_l + dh_m + 3$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	d uni-cast (1 key) + ($h_m - 1$) multi-cast (d keys) + 2 multi-cast (1 key)+ 1 multi-cast (h_l+1 keys)
Step 2		$rw + s$			1 multi-cast (s keys) + r multi-cast (w keys)	
Step 3		$h_l + q + 1$			1 uni-cast ($h_l + q + 1$ keys)	

Table 5.9: Rekeying Costs for the Member Switch Operation in Case (c)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_m + dh_l + z + 1$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	d uni-cast (1 key) + (h_m - 1) multi-cast (d keys) + (z+1) multi-cast (1 key) + 1 multi-cast (h_m+1 keys)
Step 2		$rw + s$			1 multi-cast (r keys) + r multi-cast (w keys)	
Step 3		$h_m + q + 1$			1 uni-cast (hm+q+1 keys)	

Table 5.10: Rekeying Costs for the Member Switch Operation in Case (d)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_l + dh_l + z + 1$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	d uni-cast (1 key) + ($h_m - 1$) multi-cast (d keys) + (z+1) multi-cast (1 key)+ 1 multi-cast ($h_l + 1$ keys)
Step 2		$r(w+1)$			1 multi-cast (r keys) + r multi-cast (w keys)	
Step 3		$h_l + q + 1$			1 uni-cast ($h_l + q + 1$ keys)	

Table 5.11: Rekeying Costs for the Member Switch Operation in Case (e)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_m + dh_l + d+z+2$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	d uni-cast (1 key) + h_l multi-cast (d keys) +(z+1) multi-cast (1 key) + 1 multi-cast (h_m+1 keys)
Step 2		$rw + s$			1 multi-cast (r keys) + r multi-cast (w keys)	
Step 3		$h_m + q + 1$			1 uni-cast ($h_m + q + 1$ keys)	

Table 5.12: Rekeying Costs for the Member Switch Operation in Case (f)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_l + dh_l + d+z+2$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	d uni-cast (1 key) + h_l multi-cast (d keys) +(z+1) multi-cast (1 key)+ 1 multi-cast ($h_l + 1$ keys)
Step 2		$rw + s$			1 multi-cast (r keys) + r multi-cast (w keys)	
Step 3		$h_l + q + 1$			1 uni-cast ($h_l + q + 1$ keys)	

Table 5.13: Rekeying Costs for the Member Switch Operation in Case (g)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$h_m + dh_l + dh_m + h_l+z+1$	1 uni-cast (s+1 keys)	1 multi-cast	($2d-2$) uni-cast (1 key)+1 uni-cast ($h_l + 1$ keys)+($h_m + h_l - 2$) multi-cast (d keys) +(z+1) multi-cast (1 key)+ 1 multi-cast (h_m+1 keys)
Step 2		$rw + s$			1 multi-cast (r keys) + r multi-cast (w keys)	
Step 3		h_m+q+ 1			1($h_m + q + 1$) uni-cast	

Table 5.14: Rekeying Costs for the Member Switch Operation in Case (h)

Encryption Cost				Communication Cost		
	e	GKS	CKS	e	GKS	CKS
Step 1	1		$dh_l + dh_m + 2h_l + z + 1$	1 uni-cast (s+1 keys)	1 multi-cast (1 key)	($2d-2$) uni-cast (1 key)+1 uni-cast ($h_l + 1$ keys)+($h_m + h_l - 2$) multi-cast (d keys) +(z+1) multi-cast (1 key) + 1 multi-cast (h_l+1 keys)
Step 2		$rw + s$			1 multi-cast (r keys) + r multi-cast (w keys)	
Step 3		$h_l + q + 1$			1 uni-cast ($h_l + q + 1$ keys)	

Rekeying for handoff

The rekeying for the handoff operation in MOGKM can be considered a move from a CKS to another CKS without changing the key-group. The cost of rekeying is therefore equivalent to the combined cost of a member leave and a member join operation. However, this cost does not include updating the key-group-key K_{KG_e} because no membership changes in the key-group. The cost of rekeying for a handoff is thus only the cost of step 1 in rekeying for member switch.

Periodic rekeying

Each entity group in the group manager has a different security requirement. Therefore, each entity group has a different waiting time for the periodic rekeying operation. Each periodic rekeying operation takes one encryption and sends one multi-cast message (one key).

The analysis of MOGKM shows variations in performance for the rekeying operations in different situations. Because the structure of MOGKM is divided into multiple layers and levels, the rekeying operations in MOGKM have different costs in the

different layers or levels of entities. Thus the costs of the rekeying operations (that is, member join, member leave, member switch and handoff), except periodic rekeying, have varying computation and communication costs. The variation in computation and communication costs shows the differences in performance improvement from combining multiple traditional rekeying operations of individuals into rekeying operations in MOGKM. However, the overhead from the rekeying operations across the multiple levels and layers in MOGKM may influence the total costs. The following performance analysis is used to compute the total costs of rekeying operations in MOGKM including the overhead from operations across multiple levels.

B. Performance Analysis of MOGKM

The costs for the group manager are analysed through an example of a group key with J join requests, L leave requests, S switch requests and H handoff requests. The analysis omits periodic rekeying so that MOGKM can be compared to the LKH group key management scheme. The member change rekeying operations are divided into different types as in table 5.15.

Table 5.15: Notation for Analysing Costs of MOGKM

J_1	number of join requests whereby entities are assigned into the leader-cluster
J_2	number of join requests whereby entities are assigned into a member-cluster
L_1	number of leave requests in which entities are from a member-cluster
L_2	number of leave requests in which entities are leader-candidates
L_3	number of leave requests in which entities are cluster-leaders with available candidate(s)
L_4	number of leave requests in which entities are cluster-leaders with available candidate(s)
$S_1, S_2 \ldots S_8$	number of switch requests cases 1, 2, 3, 4, 5, 6, 7 and 8
$H_1, H_2 \ldots H_8$	number of handoff requests cases 1, 2, 3, 4, 5, 6, 7 and 8

The cost of GKS in MOGKM is calculated as follows:

$$\begin{aligned} Cost_{GKS-MOGKM} &= \sum_{i=1}^{J} Cost_{join_i} + \sum_{j=1}^{L} Cost_{leave_j} + \sum_{k=1}^{S} Cost_{switch_k} \\ &= \sum_{i=1}^{J_1} Cost_{join1_i} + \sum_{i=1}^{J_2} Cost_{join2_i} + \sum_{i=1}^{4}\sum_{j=1}^{Ji} Cost_{leavei_j} \\ &\quad + \sum_{i=1}^{8}\sum_{k=1}^{Si} Cost_{switchi_k} \end{aligned} \tag{5.5}$$

By replacing $Cost_{join_1}, Cost_{join_2}, Cost_{leave_1}, \ldots, Cost_{leave_4}, Cost_{switch_1}, \ldots, Cost_{switch_7}$ and $Cost_{switch_8}$ with the encryption costs of GKS in fourteen tables 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 5.12, 5.13 and 5.14 respectively, the total encryption cost of GKS in MOGKM is rewritten as:

$$\begin{aligned} Cost_{GKS-MOGKM} =& J_1 \times (h_l + 2r + 3) + J_2 \times (h_m + 2r + 3) \\ &+ l * r(L_1 + L_2 + L_3 + L4) \\ &+ (S_1 + S_3 + S_5 + S_7) \times (r \times l + r + h_m + q + 1) \\ &+ (S_2 + S_4 + S_6 + S_8) \times (r \times l + r + h_l + q + 1). \end{aligned} \tag{5.6}$$

The total cost of $(z+1)$ CKSs in MOGKM is calculated as follows:

$$\begin{aligned} Cost_{CKSs-MOGKM} &= \sum_{i=1}^{J} Cost'_{join_i} + \sum_{j=1}^{L} Cost'_{leave_j} + \sum_{k=1}^{S} Cost'_{switch_k} + \sum_{l=1}^{H} Cost'_{handoff_l} \\ &= \sum_{i=1}^{J_1} Cost'_{join1_i} + \sum_{i=1}^{J_2} Cost'_{join2_i} + \sum_{i=1}^{4}\sum_{j=1}^{Ji} Cost'_{leavei_j} + \\ &\quad \sum_{i=1}^{8}\sum_{k=1}^{Si} Cost'_{switchi_k} + \sum_{i=1}^{8}\sum_{l=1}^{Hi} Cost'_{handoffi_l}. \end{aligned} \tag{5.7}$$

Similarly, by replacing $Cost_{join_1}, Cost_{join_2}, Cost_{leave_1}, \ldots, Cost_{leave_4}, Cost_{switch_1}, \ldots,$ $Cost_{switch_8}, Cost_{handoff_1}, \ldots, Cost_{handoff_7}$ and $Cost_{handoff_8}$ with the encryption costs of CKS in fourteen tables 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 5.12, 5.13 and

5.14, the total encryption costs of a CKS in MOGKM is rewritten as:

$$\begin{aligned} Cost_{CKSs-MOGKM} =& J_1 \times h_l + J_2 \times h_m + L_1 \times (d \times h_m + 2) + L_2 \times d \times h_l \\ &+ L_3 \times (d \times h_l + d + 3) + L_4 \times (d \times h_l + d \times h_m + h_l) \\ &+ (S_1 + H_1) \times (h_m + d \times h_m + 3) \\ &+ (S_2 + H_2) \times (h_l + d \times h_m + 3) \\ &+ (S_3 + H_3) \times d \times (d \times h_l + h_m + 1) \\ &+ (S_4 + H_4) \times (d \times h_l + h_l + 1) \\ &+ (S_5 + H_5) \times (d \times h_l + d + h_m + 3) \\ &+ (S_6 + H_6) \times (d \times h_l + d + h_l + 3) \\ &+ (S_7 + H_7) \times (d \times h_l + d \times h_m + h_l + h_m + 1) \\ &+ (S_8 + H_8) \times (d \times h_l + d \times h_m + h_l + h_l + 1). \end{aligned} \tag{5.8}$$

The average cost of a single CKS is thus equal:

$$Cost_{CKS-MOGKM} = \frac{Cost_{CKSs-MOGKM}}{z+1} \tag{5.9}$$

The total costs MOGKM are distributed over one GKS and $(z+1)$ CKSs. The GKS only handles rekeying operations in the entity layer and the key-group layer. Rekeying operations in the detail layer are managed by $(z+1)$ CKSs. However, a single rekeying operation only affects the total costs for one or two CKS(s). Therefore, the total costs of MOGKM are distributed among the GKS and CKSs as in a hybrid model (a combination of centralised and distributed approaches). The total encryption (computation) costs of GKS are shown in equation (5.6) and CKS(s) in equation (5.9). The communication costs of GKS and CKS are similar to the above computation costs. Based on the analysis, a simulation is built to compare the computation costs between MOGKM and LKH in section 5.3.3. A performance analysis for the group manager is presented in the next discussion.

C. Performance Analysis of the Group Manager

The group manager contains two separate hierarchical tree structures. One structure supports user groups while the other supports service groups. Both share the same set of key servers (one GKS and $z+1CKS$) to handle membership changes. Despite these similarities, the differences in size and character affect the performances of the two structures. Services are usually smaller in quantity and less dynamic than users. It is therefore assumed that the tree structure relating to service groups contributes fewer rekeying costs to the group manager than the tree structure relating to user groups.

Let h_l^u and h_m^u be the average heights of the leader-clusters and member-clusters in the key tree for users and let h_l^s and h_m^s be the average heights of the leader-clusters and member-clusters in the key tree for services. Let the number of member join requests in the key tree for users and the key tree for services be J_1^u, J_2^u and J_1^s, J_2^s respectively. Let the number of member leave requests in the key tree for users and the key tree for services be $L_1^u, \ldots, L_4^u$ and $J_1^s, \ldots, L_4^s$ respectively. Let the number of member switch requests in the key tree for users and the key tree for services be $S_1^u, \ldots, S_8^u$ and $S_1^s, \ldots, S_8^s$. Let the number of handoff requests in the key tree for users and the key tree for services be $H_1^u, \ldots, H_8^u$ and $H_1^s, \ldots, H_8^s$.

$$Cost_{GKS-\text{the group manager}} = Cost_{GKS-\text{MOGKM for users}} + Cost_{GKS-\text{MOGKM for services}} \tag{5.10}$$

Similarly, the total cost for $z+1CKS$ is calculated as follows:

$$Cost_{CKSs-\text{the group manager}} = Cost_{CKSs-\text{MOGKM for users}} + Cost_{CKS-\text{MOGKM for services}} \tag{5.11}$$

Because the key management layer supports the authentication controller, the group manager does not contribute to the total costs of authentication. Operations in the group manager have two main purposes: (1) handling group memberships of user

groups and service groups; and (2) securely distributing authentication keys to authorised members. The group manager also ensures that multiple replicated authentication services ASs have a unique set of identity data from user groups and service groups without operating identity database mirroring. Because the operations in the group manager are not involved in the authentication layer, they do not contribute to the total costs of authentication in the authentication controller. In the next section, the performance analysis of the authentication controller is presented.

5.2.2 Performance Analysis of the Authentication Controller

The costs of the authentication controller are derived from the authentication verification operations via the authentication protocols. The main function of the authentication controller is authentication verification via the authentication protocols using dynamic keys. An analysis of the performance of the authentication controller is thus also an analysis of the authentication protocols using dynamic keys. Dynamic keys are generated during the authentication in the proposed authentication protocols to secure authentication. The dynamic key generation scheme also contributes to the total costs of the authentication. The following subsection investigates the process to create a sequence of dynamic keys before the sequence is integrated into the authentication protocol.

A. Performance Analysis of the Dynamic Key Generation Scheme

The four steps of the dynamic key generation scheme in section 4.1.3 are summarised in the table 5.16.

Of the operations within the four steps, the key exchanges in step 1 and step 2 involve the largest communication cost. In step 1, two random keys EK and IK are generated, encrypted and sent to the involved parties. Step 2 also involves the generation, encryption and then distribution of m random initial temporary keys $TK_1, \ldots, TK_m$

Table 5.16: The Operations of the Dynamic Key Generation Scheme

Step	Operations
1	Exchange two keys EK and IK
2	Exchange m temporary key $TK_1, \ldots, TK_m$
3	Compute SK from m addition operations
4	Compute each dynamic key uses m bit-wise exclusive-ORs operations and a hashing operation

to involved parties. In order to reduce costs, these two steps are integrated into the messages of the two authentication protocols. In the ticket-based authentication protocol, step 1 is integrated into messages 2 and 3 while step 2 is integrated into message 4. Similarly, in the request-based authentication protocol, step 1 is integrated into messages 3 and 4; step 2 is integrated into message 4.

B. Performance Analysis of the Authentication Protocols

In this subsection we analyse the computation and communication costs of the authentication protocols. The computation cost includes the cost to generate nonces and keys. It does not include the costs involved in authorisation and identity look up. The computation cost includes the costs of hashing, encryption, decryption, key generation and basic operations (addition or exclusive-OR). The communication cost is simply the number of messages and their length.

B.1 Performance Analysis of the Ticket-Based Authentication Protocol

B.1.1 Computation Cost of the Ticket-Based Authentication Protocol

The analysis of the ticket-based authentication protocol can be divided into three sections based on the three involved parties (u, s and AS). u uses one hashing, two exclusive-ORs, three decryptions and one encryption in the authentication protocol. It also generates two nonces (N_u and N'_u). s uses two decryptions and two encryptions for authentication. It generates one nonce ($N_s^{'}$) and m temporary keys ($TK_1, , TK_m$).

The process to generate two dynamic keys (DK_1 and DK_2) for u and s takes $2m$ exclusive-ORs, m additions and two hashings. AS generates two keys (EK and IK). It uses one hashing and two exclusive-ORs to verify the keys and two encryptions to distribute the keys. AS requires two encryptions to create a token and an authentication ticket. The total computation costs of u, s and AS are summarised in table 5.17.

Table 5.17: Computation Cost of the Ticket-Based Authentication Protocol

	Computational Cost	Key Generation Cost
u	3 hashings, 3 decryptions (total 6+m keys), 1 encryption (1 keys), m additions and $2 + 2m$ exclusive-ORs	2 nonces
s	2 hashings, 2 decryptions (total 6 keys), 2 encryptions (total $m + 2$ keys), m additions and $2 + 2m$ exclusive-ORs	1 nonce and m keys
AS	1 hashings, 2 exclusive-ORs and 2 encryptions (total 9 keys)	2 keys

B.1.2 Communication Cost of the Ticket-Based Authentication Protocol

The ticket-based authentication protocol involves five steps and three parties (u, s and AS). In order to analyse the costs for each party, the protocol's five messages are allocated to the involved parties. u sends three messages while s and AS send one message each. These are summarised in table 5.18.

Table 5.18: Communication Cost of the Ticket-Based Authentication Protocol

	Communication Cost
u	2 messages (12 values)
s	3 messages ((m+2 values)
AS	1 messages (9 values)

B.2 Performance Analysis of the Request-Based Authentication Protocol

B.2.1 Computation Cost of the Request-Based Authentication Protocol

Similar to the ticket-based authentication protocol, the analysis of the computation cost for the request-based authentication protocol is also separated into three parts for the three different involved parties. u uses one hashing, one exclusive-OR, three decryptions and one encryption in the request-based authentication protocol. u

also generates two nonces (N_u and N'_u). Similarly, s uses one hashing, one exclusive-OR, two decryptions and two encryptions for authentication. It generates two nonces (N_s and N'_s) and m temporary keys ($TK_1, \ldots, TK_m$). The process to generate two dynamic keys (DK_1 and DK_2) for u and s takes $2m$ exclusive-ORs, m additions and two hashings. AS generates two keys (EK and IK). It uses two hashings and two exclusive-ORs to verify the message signature and two encryptions to distribute the initial keys. The total computation costs for u, s and AS are summarised in table 5.19.

Table 5.19: Computation Cost of the Request-Based Authentication Protocol

	Computation Cost	Key Generation Cost
u	3 hashings, 3 decryptions (total $5 + m$ keys), 1 encryption (2 keys), m additions and $2m + 1$ exclusive-ORs	2 nonces
s	3 hashings, 2 decryptions (total 5 keys), 2 encryptions (total $m + 2$ keys), m additions and $2m + 1$ exclusive-ORs	2 nonces and m keys
AS	2 hashings, 2 exclusive-ors and 2 encryptions (total 6 keys)	2 keys

B.2.2 Communication Cost of the Request-Based Authentication Protocol

Of the six messages in the request-based authentication protocol, u sends two messages, s sends three messages and AS sends one message. The communication cost is summarised in table 5.20.

Table 5.20: Communication Cost for the Request-Based Authentication Protocol

	Communication Cost
u	2 messages (5 values)
s	3 messages ($11 + m$ values)
AS	1 messages (6 values)

The performance analysis results in B1 and B2 are compared with the described authentication protocols in section 5.3.1. The performance comparison of the two authentication protocols in the next part is used to identify the services that are suitable for the ticket-based authentication protocol and the request-based protocol. The comparison also presents the advantages and disadvantages in terms of the resource consumption of users, services and the authentication service in each protocol.

B.3 Comparison of the Ticket-Based Authentication and the Request-Based Authentication Protocols

Each protocol has its own advantages and disadvantages in computation and communication. The computation costs of u and s of the ticket-based authentication protocol are similar to those of the request-based authentication protocol. In addition, the computation cost of AS in the ticket-based authentication protocol (1 hashing + 2 encryption (9 keys)) is larger than that of the request-based authentication protocol (2 hashings + 2 encryption (6 keys)). Therefore, the request-based authentication protocol demonstrates a computation advantage for authentication service AS. However, the communication costs of u and s in the request-based authentication protocol are larger than those of the ticket-based authentication protocol.

Each authentication protocol is suitable for different types of applications. The above analysis shows that the ticket-based authentication protocol is better suited to services running on mobile devices that have limited computation resources and low bandwidths. In this case, AS incurs higher costs so that the service has lower computation and communication costs. In contrast, the request-based authentication protocol is better suited to services running on dedicated servers in wired networks. Services require greater communication resources to send more messages. However, users u in the request-based authentication protocol utilise fewer communication resources. In the next section, the above computation and communication costs of both authentication protocols are used to calculate the total cost of the authentication controller.

B.4. Performance Analysis of the Authentication Controller

The total cost of the authentication controller is the combined cost of authentication from the two authentication protocols. Let A_1 and A_2 be the number of authentication requests using the ticket-based authentication protocol and the request-based

authentication protocol respectively. The total costs of the authentication controller are summarised in table 5.21.

Table 5.21: Total Costs for an Authentication in the Authentication Controller

	Computation Cost	Communication Cost
u	$A_1(10+m)+A_2(10+m)$	$3A_1+2A_2$
s	$A_1(8+m)+A_2(10+m)$	A_1+3A_2
AS	$10A_1+8A_2$	A_1+A_2

The performance analysis in table 5.21 is used to conduct a simulation to compare AMUS with existing authentication approaches in section 5.3.1. The security and performance analysis of the proposed authentication realisation in section 5.1 and 5.2 are compared and discussed further in the next section.

5.3 Discussion

This section investigates the analysis results of the previous sections to examine in more detail the security and performance features of the authentication realisation and its mechanisms. The results are also compared with previous approaches. The investigation determines that the proposed authentication can achieve both security and efficiency. It also confirms the advantages of the authentication realisation over previous approaches.

In the first subsection, a comparison of the security and performance features between the authentication realisation and other authentication methods (Kerberos, OpenID and PAKE) is presented. The ability to overcome the security vulnerabilities found in section 2.7 is used as the criteria to examine and to compare the security features. The comparison results are used to determine whether the authentication realisation can achieve the security features necessary to resist attacks. Likewise, a performance comparison using computation and communication cost analysis, queueing theory and simulation is given in section 5.3.1. The comparison results are used to

confirm the advantages in both security and performance of the authentication realisation over other authentication approaches. The following section, 5.3.2, compares the performance of MOGKM to LKH using queueing theory and simulation. Section 5.3.3 discusses how to meet security requirements while incurring minimal costs in dynamic keys and authentication. An extended authentication model for audit tracing is proposed in section 5.3.4.

5.3.1 A Comparison of the Proposed Authentication Protocols with Existing Authentication Protocols

In this section we compare two features: security and efficiency. The comparison validates the balance between the two goals of security and efficiency. For an authentication method for wireless network users and services to be effective, it is critical that the method balances security and efficiency. Of the two goals, security is the most important.

A. Security

In this section, we examine and compare three existing authentication methods (Kerberos, OpenID and PAKE) (see section 2.7), the ticket-based authentication protocol (TBAP) and the request-based authentication protocol (RBAP) in section 4.4 in light of the three common attacks (replay, cryptanalysis and phishing). The results are listed in table 5.22.

Table 5.22: Security Comparison of Possible Attacks on the Existing Authentication Methods and the Two Proposed Authentication Protocols in the Authentication Realisation

	Replay	**Cryptanalysis**	**Phishing**
Kerberos	Possible	Possible	Secure
OpenID	Secure	Possible	Possible
PAKE	Secure	Possible	Secure
TBAP	Secure	Secure	Secure
RBAP	Secure	Secure	Secure

The comparison in table 5.22 shows that only the ticket-based and request-based authentication protocols are secure under these three common attacks on authentication. As the proposed authentication is based on group key management and dynamic keys, the proposed authentication realisation with the two authentication protocols is still secure even if one or more dynamic keys or authentication keys are compromised. In contrast, compromised authentication keys or session keys in the three authentication methods may lead to unauthorised access even with the password-based public key cryptographic authentication method (PAKE). Hence, we can conclude that the authentication realisation can achieve security via its two authentication protocols.

B. Performance

We examine the performance of the different authentication methods by two different methods: (1) analysing the total number of encryptions and number of messages; and (2) analysing the response time of the authentication realisation using queueing theory and simulation. The first method is based on an analysis of each of the two authentication protocols in the authentication realisation while the second method is conducted based on the combination of the two authentication protocols as a whole. The results of the comparison using the second method via simulation confirm the comparison results based on the number of encryptions and messages.

B.1 Computation and Communication Comparison

We compare the results from the existing authentication methods in section 2.7 against the performance analysis results of the ticket-based authentication and the request-based authentication protocols found in section 5.2.2. Two temporary keys (m=2) in both the ticket-based and the request-based authentication protocols are used for the comparison. The results are shown in table 5.23.

The results in table 5.23 show that the request-based authentication protocol (RBAP) and the ticket-based authentication protocol (TBAP) are more efficient than the other

Table 5.23: Performance Comparison of the Proposed Authentication Protocols and the Existing Authentication Methods

	Asymmetric Encryption	**Symmetric Encryption**	**Number of Messages**
Kerberos	0	23	6
OpenID	6	62	25
EKE	4	12	5
TBAP	0	15	5
RBAP	0	14	6

authentication methods. The request-based and ticket-based authentication protocols show clear advantages in terms of computation cost as they incur the least computation cost for encryption. Furthermore, the communication cost of these two proposed authentication protocols is no larger than the other protocols.

In the next section, to determine whether the authentication realisation meets the goal of efficiency, the performance of the authentication realisation as a whole is compared to the performance of the Kerberos, OpenID and PAKE approaches when using queueing theory and simulation.

B.2 Comparison using Queueing Theory and Simulation

In this section, queueing theory [GC98] is used to analyse the performance of the authentication realisation. Queueing systems with autonomous services were originally studied in [Bor76] [Bor84] for modelling systems in maths. They allow several performance calculations including the average waiting time in a queue and the average response time. Of the two types of queueing networks (open queueing networks and closed queueing networks), the closed queueing network is more suitable because it has a finite number of users (or requests).

We developed a modelling strategy using closed queueing networks [Kum95] to represent the authentication realisation with one AS and $T_n + R_n$ services in wireless networks. Let T_n services using the ticket-based authentication protocol be $S_{T1}, \ldots, S_{Tn}$ and let R_n services using the request-based authentication protocol be $S_{R_1}, \ldots, S_{R_n}$.

This technique allows us to model the authentication realisation with a combinations of T_n services using the ticket-based authentication protocol and R_n services using the request-based authentication protocol. Figure 5.1 presents the queueing network topology of the authentication realisation. The queueing network maintains independent Markov chains with Poisson arrivals for $T_n + R_n$ services. Users travel in the closed queueing network between services and sequentially wait to be served.

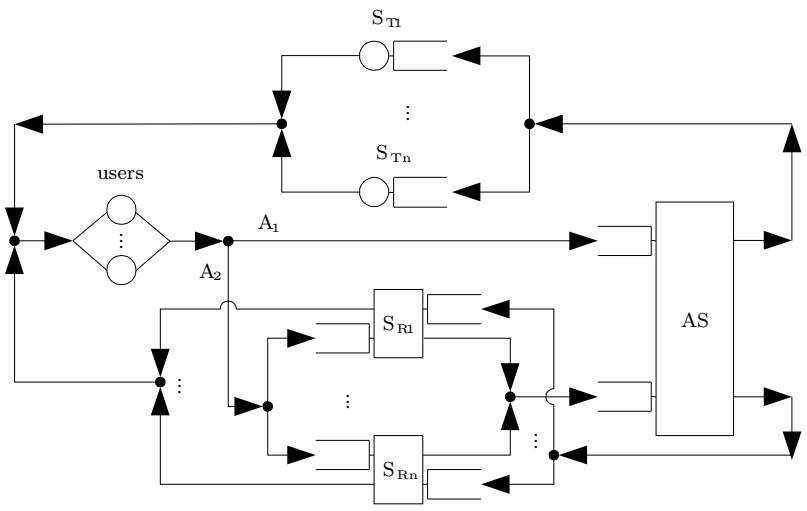


Figure 5.1: The Queueing Network Topology for the Authentication Realisation

The performance measurements are derived from the queueing network for each service and the authentication realisation as a whole system. The measurements include the average load (the average number of authentication requests), the average waiting time, the authentication throughput and the response times. The response times obtained from the measurement of the queueing network for the authentication realisation are compared with those of Kerberos, OpenID and PAKE.

Figure 5.2 plots the authentication load against the response times of the authentication realisation for the AMUS model, Kerberos, OpenID and PAKE (we also built

queueing networks for these authentication methods). The service times for a single block symmetric encryption/decryption, hash function and a asymmetric encryption/decryption in both services were assumed to be $19 * 10^{-5}ms, 8.6 * 10^{-5}ms$ and $0.35ms$ respectively in a measurement running on a Intel Pentium 4 (Prescott) 2.66Ghz. The service times of these cryptographic operations in mobile devices were assumed to be $0.2ms, 0.9ms$ and $40ms$. The wireless network speed was assumed to be 1,024 kbps in the simulation runs for the authentication realisation, Kerberos, OpenID and PAKE.

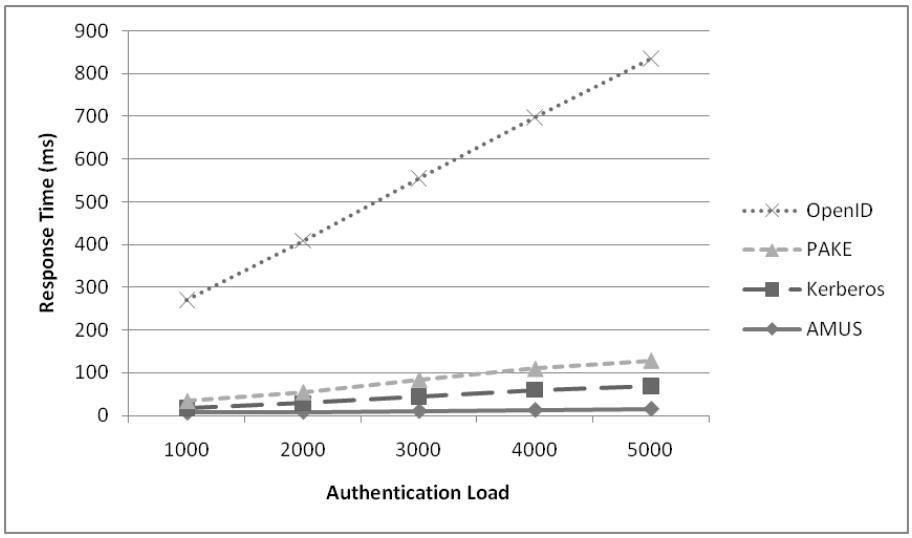


Figure 5.2: The Response Time versus Authentication Load Comparison

The response times in figure 5.2 rise from the initial point to demonstrate that the response times of the authentication realisation of the AMUS model are the smallest of all the protocols. Because of the high computation cost associated with using asymmetric key cryptography on mobile devices, the response times for the OpenID and PAKE authentication methods are the highest. Kerberos also has longer response times than the proposed authentication. In conclusion, the use of queueing theory demonstrates the superiority, in terms of performance, of the realisation authentication proposed in the previous section.

In the authentication layer, the advantages in both security and performance of the authentication in wireless networks are achieved by using dynamic key cryptography. In the above comparisons, the authentication realisation is assumed to use two temporary keys (m = 2) to create dynamic keys. However, the authentication realisation is not restricted to only two temporary keys for balancing security and efficiency. The number of temporary keys used to create dynamic keys can be increased to enhance security. The following discussion investigates parameter variations in dynamic keys, the effects on security and performance and the weakness of dynamic keys.

5.3.2 Discussion of the Dynamic Key Cryptography

In this subsection we examine features of the dynamic key cryptography used for authentication. The first part (A) answers the question of how to find the smallest key size and number of parameters (m) for function $f(.)$ to satisfy the security requirements. The second part (B) investigates how to find the most appropriate length of the dynamic key sequence. The last part (C) discusses the drawbacks of dynamic keys and in particular, synchronisation problems and how to reduce these problems and resynchronise dynamic keys.

A. Key Size and Parameters

Cryptographic systems usually require longer key sizes to achieve higher security. In dynamic key cryptography systems, the security of dynamic key sequences also depends on the key sizes and the number of parameters of function $f(.)$ to generate dynamic keys as analysed below. To prove this statement, the following theorem is introduced.

Theorem 5.4. *The security of a cryptography system using a dynamic key sequence is proportional to the number of parameters m for function f(.) used to create dynamic keys and the key size of each dynamic key.*

Proof. Let A be the algorithm to guess a dynamic key. Let the probability of correctly guessing a dynamic key DK_i be denoted as $Pr(A(DK_i))$. To compute the current and future dynamic keys $DK_c, DK_{c+1}, \ldots$, an adversary must obtain the function $f(.)$ as well as its parameters.

The function $f(.)$ used to generate the current dynamic key DK_n from $m+1$ input parameters in the dynamic key generation scheme is written as follows:

$$DK_n = f(SK, DK_{n-m}, \ldots, DK_{n-2}, DK_{n-1}) \tag{5.12}$$

Because of property 4.2 and lemma 4.2, the probability of guessing one dynamic key correctly is assumed to be independent from an adversary's knowledge of any used dynamic key. The security level for a dynamic key cryptographic system is presented as the probability of breaking the sequence of dynamic keys. Assuming that an adversary knows the function $f(.)$, the probability of breaking the sequence of dynamic keys can be computed as:

$$Pr(\{DK_i\}) = Pr(A(SK)) \times Pr(A(DK_{i-m})) \times \ldots \times Pr(A(DK_{i-1})) \tag{5.13}$$

If the cryptographic algorithm can only be broken by an exhaustive search (that is, a brute-force attack), the probability of correctly guessing DK_{i-1} is $\frac{1}{2^s}$, with s being the bit length of the dynamic key DK_i. Where $m+1$ is the number of parameters of function $f(.)$, the probability of breaking the sequence of dynamic keys can be rewritten as follows:

$$Pr(\{DK_i\}) = \frac{1}{2^s} \times \overbrace{\frac{1}{2^s} \times \ldots \times \frac{1}{2^s}}^{m \text{ times}} = \frac{1}{2^{ms+s}} \tag{5.14}$$

This result shows that the security of symmetric cryptography using dynamic keys is proportional to both m and s. The probability of breaking the cryptography

system is reduced significantly when either m or s is increased. While the probability of breaking a normal session key symmetric cryptographic system is $\frac{1}{2^s}$, the probability of breaking a symmetric dynamic key cryptographic system is reduced to $\frac{1}{2^{ms+s}}$. □

The equation (5.14) shows two methods to increase the security of the sequence of dynamic keys. The first method involves increasing the key size (s) of a dynamic key while the second method involves increasing the number of parameters (m) used to create the dynamic keys (starting with $TK_1 \ldots TK_m$).

However, both of these methods also have disadvantages in terms of high storage, computation and communication costs:

i. Increasing the key size of the dynamic keys incurs additional encryption and decryption, which in turn often consumes more computational resources. The larger key size also requires slightly more storage space for the longer key size in memory.

ii. Increasing the number of parameters for function $f(.)$ will use more memory to ensure all parties remember these parameters. This solution may not suit mobile devices with limited memory and battery power.

Instead of targeting strongly secure dynamic key sequences for all systems, we try to achieve the smallest key size and number of parameters of functions $f(.)$ while still maintaining the security level for the cryptographic system.

Corollary 5.3. *The minimum value of (ms +s) to keep the probability of breaking the dynamic key sequence larger than r is $log_2 r$.*

Proof. Let $r(r < \frac{1}{2^s}, r \in [0,1])$ be the highest acceptable probability of breaking the dynamic key sequence, or $Pr(\{DK_i\}) \leq r$. The equation (5.14) is deduced as follows:

$$\frac{1}{2^{ms+s}} \leq r$$
$$\text{or } ms + s \geq log_2 \frac{1}{r} = -log_2 r \tag{5.15}$$

□

Figure 5.3 describes the relationship between the requirement of value $ms+s$ and the value of r.

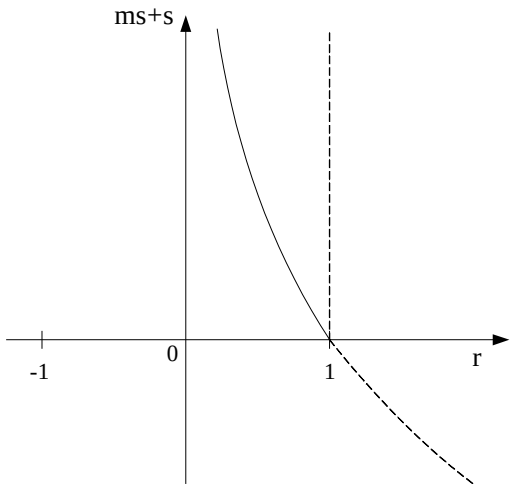


Figure 5.3: The Relationship Between the Value $ms + s$ and the Value of r

For example, a dynamic key cryptographic system has a 256 bit key size. The highest acceptable probability of breaking the dynamic key sequence r is $\frac{1}{2^{1024}}$ (which is equivalent to 1024 bit key cryptography) with the number of parameters m being $m \geq 3$. In other words, the security of a dynamic key cryptographic system using a 256 bit key size and remembering three parameters is equivalent to that of a symmetric cryptographic system using 1024 bit key size without sacrificing computational performance for encryption.

As shown in the analysis above, the sequence of dynamic keys is not secure on an on-going basis. After a period of time, the probability of breaking the dynamic key sequence may be higher than the security requirement level. A new dynamic key

sequence needs to be generated using the repeated dynamic key generation scheme to replace the old dynamic key sequence. The following analysis is used to determine the point at which to generate a new dynamic key sequence.

B. Dynamic Key Sequence Length

A dynamic key sequence is only secure for a limited period of time. As in the analysis of theorem 5.4, the probability of breaking the sequence of dynamic keys is $Pr(\{DK_i\}) = \frac{1}{2^{ms+s}}$. With enough computational power and time, adversaries are able to guess the sequence of dynamic keys. To prevent this vulnerability, once the probability of breaking the dynamic key sequence is higher than the level of security required for the system, the dynamic key sequence is replaced by a new dynamic key sequence using the repeat dynamic key generation scheme.

Theorem 5.5. *A secure dynamic key sequence has no more than* $\frac{y2^{m+s}}{x} - \frac{y}{xr}$ *keys, where y is the rate of using dynamic keys, x is the trial rate to break a dynamic key sequence and r is the maximum allowed probability of breaking a dynamic key sequence.*

Proof. Let x be the average possible number of trials that an adversary can perform to break the dynamic key sequence. At the time t, xt trials are produced. The probability of the adversary correctly guessing the dynamic key sequence is rewritten as follows:

$$Pr(\{DK_i\}) = \frac{1}{2^{ms+s} - xt} \tag{5.16}$$

In the security requirement for the system, let r ($\frac{1}{2^{ms+s}} < r < \frac{1}{2^s}$) be the maximum probability of breaking the dynamic key sequence in a matter of time and let y be the average number of dynamic keys used in a unit of time. The condition necessary to maintain the security of the dynamic key sequence is written as follows:

$$\begin{aligned} Pr(\{DK_i\}) = \frac{1}{2^{ms+s} - xt} \leq r \\ \text{or } t \leq \frac{2^{ms+s}}{x} - \frac{1}{xr} \end{aligned} \tag{5.17}$$

Let n be the number of dynamic keys have been used in the sequence, or $n = ty$. We then have:

$$n \leq \frac{y2^{m+s}}{x} - \frac{y}{xr} \tag{5.18}$$

Thus, the highest number of secure dynamic keys in the sequence n is $\frac{y2^{m+s}}{x} - \frac{y}{xr}$. When the length of the sequence is higher than this value, the probability of breaking the sequence is larger than r. □

For example, let x be $2^{64}, y = 2^{24}, m = 3, s = 256$, and r be $\frac{1}{2^{1020}}(\frac{1}{2^{1024}} < r < \frac{1}{2^{256}})$. n should be smaller than 15×2^{980} in order to keep the dynamic key sequence secure. If n is larger, the repeat dynamic key generation scheme is invoked to generate a new dynamic key sequence.

C. The Synchronisation Problem

Although it has security advantages, dynamic key cryptography has a major drawback known as the synchronisation problem. Based on symmetric cryptography, in a dynamic key cryptography system, the dynamic keys of senders and receivers need to be identical. If the dynamic keys between senders and receivers are not the same, communication breaks down because receivers are no longer able to decrypt messages from senders.

The synchronisation problem results, among other things, from connection problems and malicious attacks from adversaries. Synchronisation problems can arise because of connection problems; communication interruptions are inherent to wireless networks. A synchronisation problem can also happen when an adversary breaks cryptography using a current dynamic key sequence. In another scenario of malicious attack, an adversary can masquerade as a valid sender and send messages encrypted by invalid dynamic keys to confuse a receiver. The adversary can also launch a "man in the middle attack" to interfere with or intercept the communication. If either the

receiver or the sender is unaware of the corrupted communication messages, the dynamic keys of the receiver or sender are discarded after being used to decrypt the corrupted messages. The receiver then is no longer able to decrypt future valid messages because its dynamic keys do not match those of the sender.

The risk of the synchronisation problem occurring due to malicious interference can be reduced by appending a message authentication code. A message authentication code is a piece of information normally generated by hashing a message and then adding the hashed message into the original message. The code can be used to authenticate the source and the data integrity of the message. By validating the message authentication code, fraudulent messages from masquerading senders can be detected and ignored by receivers. The order of dynamic keys used between senders and receivers can thus be preserved. However, this approach may create overheads for creating and validating message authentication code that reduce the performance of the cryptography. In addition, message authentication codes are not able to prevent synchronisation problems caused by broken connections or compromised dynamic key sequences.

When a synchronisation problem happens, the resynchronisation dynamic key scheme is invoked to generate a new dynamic key sequence. For example, sender Alice notices that communication with receiver Bob is broken because of a synchronisation problem. Alice sends Bob a resynchronisation message. Bob sends an acknowledgement to Alice and starts the resynchronised dynamic key scheme as follows:

Resynchronised Dynamic Key Scheme

Step 1: Both Alice and Bob calculate a new initial key $EK^{'}$ from the session keys in the old sequence:

$$EK^{'} = SK \oplus EK \oplus IK \tag{5.19}$$

Step 2: Alice generates a new set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ and sends these to Bob encrypted by $EK^{'}$:

$$A \rightarrow B : \{TK_1^{'}, \ldots, TK_m^{'}\}EK^{'}, h(TK_1 \oplus \ldots, TK_m \oplus EK^{'}) \tag{5.20}$$

Step 3: Both Alice and Bob compute a new seed key $SK^{'}$ from SK and $TK_1^{'}, \ldots TK_1^{'}$ using additions:

$$SK^{'} = SK + TK_1 + \ldots TK_2 \tag{5.21}$$

Steps 4: Both Bob and Alice generate the sequence of dynamic keys.

This step is the same as step 4 in the initial dynamic key generation scheme. The new seed key $SK^{'}$ and the new set of temporary keys $TK_1^{'}, \ldots, TK_m^{'}$ are used to calculate a new sequence of dynamic keys $TK_1^{'}, \ldots, TK_m^{'}$.

In addition to the security and performance advantages contributed by dynamic key cryptography in the authentication layer, the authentication realisation also relies on the security and efficiency of the MOGKM scheme used by the group manager in the key management layer. The security of MOGKM is confirmed in section 5.1.1. In the next section, a performance comparison between MOGKM and LKH is presented to demonstrate the performance advantage of the group manager.

5.3.3 Comparison Between the MOGKM and LKH

To analyse the performance of MOGKM, we built another model using closed queueing networks. The model contains one centralised GKS and a $z+1$ distributed cell key server CKSs. Each GKS and each CKS are a modelled M/M/1 system. Rekeying operations such as member join, member leave, member switch and handoff are assumed to be queued at GKS as a Poisson process with constant rate $\lambda_{join}, \lambda_{leave}, \lambda_{switch}$ and $\lambda_{handoff}$. Figure 5.4 presents the queueing network topology of MOGKM.

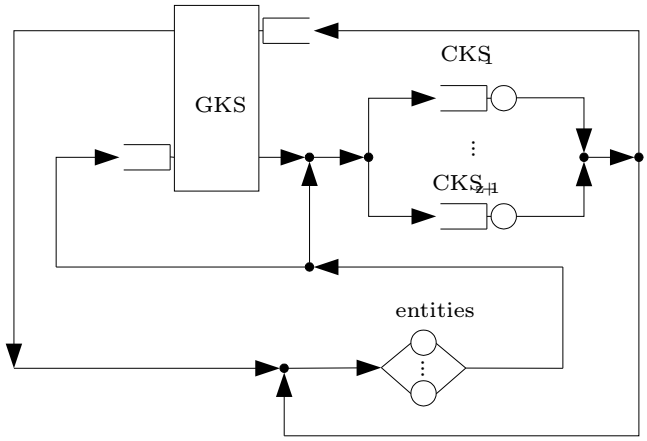


Figure 5.4: The Queueing Network Topology for MOGKM

The response time of MOGKM, derived from the queueing network, is compared to that of LKH. Figure 5.5 plots the response time versus the user load of MOGKM and LKH. The simulation used in the measurement has one GKS and 3 $CKSs$ ($z = 2$).

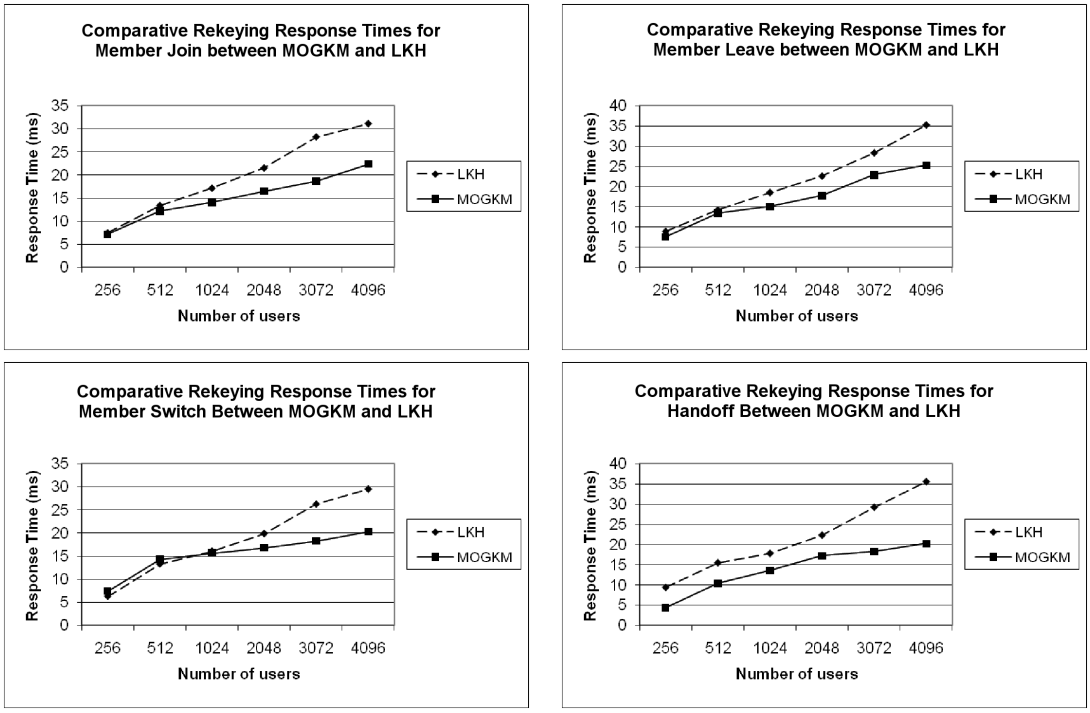


Figure 5.5: Comparison of Response Times of Rekeying Operations Between LKH and MOGKM

The comparison results in figure 5.5 show the performance advantages of MOGKM over LKH. The response times of the member join, member leave and handoff rekeying operations of MOGKM are smaller than those of LKH. Only in rekeying for member switch with less than 1024 entities, the response time of LKH's rekeying operations is smaller than that of MOGKM. However, when the number of entities is larger than 1000, the rekeying for member switch of MOKGM is faster than that of LKH.

In the AMUS authentication model, authentication for wireless network users and services is conducted via their group identities. The use of group identities helps to enhance the scalability of the authentication. However, some services require individual identity authentication for audit tracing. In the next section, an extension of the AMUS authentication model performing individual and group authentication together is presented.

5.3.4 An Extension for Authentication with Audit Tracing

The AMUS authentication model can be extended to perform individual authentication for audit tracing. In the extension, the user's individual identity and its user group identity are verified together in the authentication verification. The identity verification combination renders the authentication extension similar to two-factor authentication [Sch05]. The extended authentication protocol differs slightly from the request-based authentication protocol. The ticket-based authentication protocol can also be modified to have both individual and group authentication. The extended protocol based on the request-based authentication protocol is formalised as follows:

1. $u \rightarrow\ s : u, UG, N_u, h(N_u \oplus K_{UG}^{auth} \oplus DK_i^u)$
2. $s \rightarrow\ AS : u, UG, N_u, h(N_u \oplus K_{UG}^{auth} \oplus DK_i^u), SG, N_s, h(N_s \oplus K_{SG}^{auth})$
3. $AS \rightarrow\ s : \{\ EK\}K_{i+1}^u, \{\ IK, N_u\}K_{UG}^{auth}, \{EK, IK, N_s\}K_{SG}^{auth}$
4. $s \rightarrow\ u : \{\ EK\}K_{i+1}^u, \{\ IK, N_u\}K_{UG}^{auth}, \{TK_1, \ldots, TK_m, N_s'\}EK$
5. $u \rightarrow\ s : \{N_s' + 1, N_u'\}DK_1$

6. $s \rightarrow u : \{N'_u - 1\}DK_2$

In the extended protocol, both the individual dynamic key DK_i^u and the user group authentication key K_{UG}^{auth} are used for authentication. In messages 1 and 2, u specifies that it wants to authenticate the identities of both u and UG. In message 3, EK is encrypted by DK_i^u and IK is encrypted by K_{UG}^{auth}. Lacking both DK_i^u and K_{UG}^{auth}, u is not able to decrypt EK and IK to authenticate with s to access the service. In other words, u needs to be able to use both the authentication keys (DK_i^u and K_{UG}^{auth}) for authentication in the extended authentication protocol. The rest of the protocol remains the same as the request-based authentication protocol.

5.4 Summary

In this chapter, the authentication realisation presented in the previous chapter was analysed and discussed in terms of security and efficiency. In the analysis of security, each component, including the group manager and the authentication controller, was formally analysed. Later, the security of the combination of all the components was verified as an entire authentication. Because the key management layer is transparent to the authentication layer, the performance of the group manager does not affect the performance of the authentication controller. The performance of the group manager and the authentication controller can thus be analysed separately.

The security analysis confirms that the proposed realisation using dynamic keys and MOGKM can provide secure authentication to resist possible attacks. The formal analysis in sections 5.1.1, 5.1.2 and 5.1.3 shows that MOGKM and both the ticket-based and request-based authentication protocols using dynamic keys meet their security goals. The validation results of the three possible types of attacks corroborate that the authentication controller is secure under replay, phishing and cryptanalysis attacks. The group manager using MOGKM only distributes authentication keys to

valid users and services having authorised group memberships. Likewise, authentication via the authentication controller only allows authorised users to authenticate with, and access, valid services. Finally, the analysis results of the combination of the group manager and the authentication controller prove that the authentication realisation is secure.

The performance analysis shows that the authentication realisation is efficient in wireless networks. In the performance analysis for MOGKM, the cost of the rekeying operations is distributed throughout the hybrid structure by using one GKS and many CKS. The total cost for the rekeying of GKS and CKS is found in equations (5.6) and (5.8) and the simulation in section 5.3.3. The ticket-based authentication protocol uses 17 encryptions and 5 messages; the request-based authentication uses 16 encryptions and 6 messages. Each of these two protocols has its own advantages and is suitable for different types of services.

A number of comparisons and discussions were conducted in section 5.3. The comparison in section 5.3.1 showed that the authentication realisation based on dynamic key cryptography and MOGKM has better security and performance features than the three existing authentication methods of Kerberos, OpenID and PAKE. The security and performance features of dynamic key cryptography were discussed in section 5.3.2. The performance comparison between MOGKM and LKH in section 5.3.3 demonstrates that MOGKM is more efficient than LKH in wireless networks.

Finally, an extension of the AMUS authentication model for audit tracing was proposed in section 5.3.4. The extended authentication model supports both individual and group authentication so that services can obtain an individual user identity for auditing tasks as well as a group user identity for authentication and authorisation. The authentication in the extended authentication model is slightly different from the original request-based authentication protocol, but the extension only slightly increases computation and communication costs.

This chapter has demonstrated the effectiveness of the AMUS authentication in terms of security and efficiency and thus its suitability for use with mobile devices in wireless networks. The next chapter summarises the qualities of AMUS, the contributions made by this research, potential further research and concludes the thesis.

Chapter 6

Conclusion

The number of users and services of mobile devices in wireless networks is constantly increasing. Mobile users require secure access to services. The emergence of cloud computing for certain groups of users has also added to the need for strong security. Secure access can be achieved through an authentication model that is characterised by security, flexibility, efficiency and scalability. However, existing approaches lack these properties; they provide authentication for only a limited number of services and users and have drawbacks such as low levels of security due to static shared keys or suffer inefficiencies when groups of users are involved. To overcome the limitations of existing approaches, this thesis has proposed a new authentication model - AMUS - that possesses the required properties.

The AMUS model's *security* protects users and services from unauthorised access and resists common attacks. The idea of deriving a strong dynamic key scheme and applying it together with good group key management in the proposed model (described in chapter 4) not only makes common attacks less possible but also allows the secure sharing of services.

The AMUS model's *efficiency* minimises computation, communication and management costs. In order to achieve the low costs without compromising security, the

dynamic key scheme in section 4.1.3 is employed. The dynamic key scheme enhances security without compromising efficiency. The proposed authentication realisation (chapter 4) and its analysis (chapter 5) demonstrate the model's efficiency.

The AMUS model's *flexibility* allows different cryptographic and group communication mechanisms to be employed to suit different mobile devices and situations. This flexibility is achieved by creating and separating two layers of architecture in the proposed authentication (chapter 3). Implementations can then use different mechanisms in each layer.

The AMUS model's *scalability* enables it to adapt to rapid changes in the number and types of users and services so that it is suitable for both small and large-scale wireless networks. To achieve scalability, the proposed authentication model organises users and services into user groups and service groups (described in chapter 3). To manage group memberships, the group manager of the model in section 3.2.1 and membership-oriented group key management for wireless networks in section 4.2 are employed.

6.1 The Research Contribution

The research in this thesis contributes to future developments in knowledge concerning the prevention of unauthorised access for dynamic wireless network users and services. Its contributions include the following:

- a novel authentication model for wireless network users and services (proposed in chapter 3). The model contains basic elements (users, services, user groups, service groups and identities), a collection of relationships between elements, a group manager and an authentication controller. The group manager handles the management tasks in authentication while the authentication controller conducts authentication tasks. Using these components, the proposed model

reduces management costs and simplifies authentication and authorisation. Unlike existing models, the AMUS model can achieve both structural and load scalability in authentication for large wireless network services and mobile users when under heavy use.

- an authentication architecture (described in chapter 3) derived from the proposed authentication model. The architecture has two layers: the key management layer and the authentication controller layer. These provide a guideline for any realisation of the proposed authentication model. As a result of the two-layer architecture, different mechanisms can be employed in the authentication model according to the situation. The authentication model can thus achieve both flexibility and extendibility. An extension of the authentication model using both individual and group authentication for audit tracing was presented in section 5.4.

- a dynamic key scheme (chapter 4) to support the proposed authentication. The scheme can be used to generate one-time symmetric cryptography keys. It is based on parameters that determine the security level of the dynamic key sequence. Unlike traditional cryptography which relies on key exchange schemes, dynamic keys are generated offline by the dynamic key generation scheme (see section 4.1.3). Dynamic key cryptography has security and efficiency advantages over both traditional long-term shared key and session key cryptography. The produced dynamic keys can help authentication resist replay attacks and reduce cryptanalysis risks (see theorem 5.2 and property 5.1).

- a group key management scheme drawn from the hybrid group key management approach to support authentication in wireless networks (described in chapter 4). The secure group manager (see property 5.3) is used to perform key management and to manage dynamic group memberships and key distributions for wireless network users and services. By dividing the logical structure of the

original group key tree into three different layers (the entity layer, the key-group layer and the detail layer), the key tree can support scalable multi-membership entities. The work load of group membership management is distributed on a hybrid structure having one GKS and multiple CKSs. In addition to supporting authentication, the group key manager is also used to secure group communications.

- a dynamic group-based authentication derived from the AMUS authentication model (described in chapter 4). The group key management scheme is utilised in the group manager and dynamic key cryptography is adapted for authentication protocols in the authentication controller. The analysis in chapter 5 found that authentication based on dynamic key cryptography, group key management and the AMUS authentication model achieves strong security and high efficiency, as follows:

  - *Security*. Theorem 5.3 proves that the proposed authentication achieves strong authentication for users and services in wireless networks. The security analysis in section 5.3.1 shows that AMUS authentication is the only model that can resist common attacks that compromise other authentication methods (that is, Kerberos, OpenID and PAKE).

  - *Efficiency*. The analysis (see section 5.2) and comparisons (see section 5.3.1) show the performance advantage of AMUS authentication. AMUS authentication has the lowest computation cost (72% of Kerberos, 0.15% of OpenID and 0.4% of PAKE). It also has lowest communication cost (5 to 6 messages compared to 6 messages in Kerberos, 25 messages in OpenID and 5 messages in PAKE). Authentication services in AMUS also have lower storage and management costs.

In summary, the dynamic group-based authentication derived from AMUS authentication is unique in that it provides strong, secure, efficient, flexible and scalable authentication compared to other authentication approaches for wireless network users and services.

6.2 Future Work

Our research has achieved a number of goals and has also provided us with the opportunity to consider two related future research ideas. To date, little consideration has been given to how best to group users and services with the aim of promoting stronger authentication with higher efficiency. We do not know if classifying users and services for authentication can further improve efficiency. It would also be desirable to be able to integrate authentication and authorisation to provide stronger access controls. In our proposed model, the application of dynamic key theory and group key management ensured strong security. However, we would like to investigate the possibility of integrating authentication and authorisation in order to obtain better access controls for wireless network users and services.

In conclusion, this thesis has demonstrated that the proposed authentication model overcomes the problems of existing authentication approaches and can provide secure, efficient, flexible and scalable authentication for wireless network users and services. It motivates further research in this area in order to protect wireless network users and services and to increase scientific knowledge.

Bibliography

[AG03] Alessandro Aldini and Roberto Gorrieri. A study about trade-off between performance and security in an internet audio mechanism. In *IST/FET International Workshop of Global Computing. Programming Environments, Languages, Security, and Analysis of Systems*, volume 2874 of *Lecture Notes in Computer Science*, pages 203–228, 2003.

[AGG+09] L. Arnone, C. Gonzalez, C. Gayoso, J.Castineira Moreira, and M.Liberatori. Security and BER performance trade-off in wireless communication system applications. *Latin American Applied Research*, 39(3):187–192, 2009.

[AGIS05] Frank Adelstein, Sandeep K.S. Gupta, Golden G. Richard III, and Loren Schwiebert. *Fundamentals of Mobile and Pervasive Computing*. McGraw-Hill Professional Engineering, 2005.

[AR07] Hamad Alrimeih and Daler Rakhmatov. Security-performance trade-offs in embedded systems using flexible ECC hardware. *IEEE Design & Test*, 24(6):556–569, 2007.

[AST00] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. New multiparty authentication services and key agreement protocols. *IEEE Journal on Selected Areas in Communications*, 18(4):628–639, 2000.

[AT91] Martin Abadi and Mark R. Tuttle. A semantics for a logic of authentication. In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, pages 201–216, 1991.

[BAN89] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *Operating System Review*, 23(5):1–13, 1989.

[BAN90] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[Bar06] Gregory V. Bard. A challenging but feasible blockwise-adaptive chosen-plaintext attack on SSL. In *Proceedings of the International Conference on Security and Cryptography SECRYPT*, pages 7–10. INSTICC Press, 2006.

[BCEP04] Emmanuel Bresson, Olivier Chevassut, Abdelilah Essiari, and David Pointcheval. Mutual authentication and group key agreement for low-power mobile devices. *Journal of Computer Communications*, 27(17):1730–1737, 2004.

[BCS+99] Deepak Bansal, Anurag Chandra, Rajeev Shorey, Ashutosh Kulshreshtha, and Manish Gupta. Mobility models for cellular systems: cryptography and handoff probability. In *Proceeding of IEEE 49th of Vehicular Technology Conference*, volume 3, pages 1794–1798, 1999.

[BDR+96] Matt Blaze, Whitfield Diffie, Ronald L. Rivest, Bruce Schneier, Tsutomu Shimomura, Eric Thompson, and Michael Wiener. Minimal key lengths for symmetric ciphers to provide adequate commercial security. Technical report, A report by an ad hoc panel of cryptographers and computer scientists, 1996. Retrieved 15 August, 2009, from `http://www.crypto.com/papers`.

[Bel08] Michael Bell. *Introduction to Service-Oriented Modelling: Service Analysis, Design, and Architecture*. Wiley & Sons, 2008.

[BFMT03] Michele Bugliesi, Riccardo Focardi, Matteo Maffei, and Fabio Tudone. Principles for entity authentication. In *Proceedings of the 5th International Conference Perspectives of System Informatics*, volume 2890 of *Lecture Notes in Computer Science*, pages 294–307, 2003.

[BK98] Alex Biryukov and Eyal Kushilevitz. From differential cryptanalysis to ciphertext-only attacks. In *Advances in Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 72–88, 1998.

[BLB+00] Keith Bennett, Paul Layzell, David Budgen, Pearl Brereton, Linda Macaulay, and Malcolm Munro. Service-based software: The future for flexible software. In *Proceedings of the Seventh Asia-Pacific Software Engineering Conference*, 214-221, 2000.

[BM91] Steven M. Bellovin and Michael Merritt. Limitations of the kerberos protocol. In *Winter 1991 USENIX Conference Proceedings*, pages 253–267, 1991.

[BM92] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 72–84, 1992.

[BM93] Steven M. Bellovin and Michael Merritt. Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password file compromise. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 244–250, 1993.

[BM03] Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 2003.

[Bon00] Andr B. Bondi. Characteristics of scalability and their impact on performance. In *Proceedings of the 2nd international workshop on Software and performance*, pages 195 – 203, 2000.

[Bor76] Alexandr Alekseevich Borovkov. *Stochastic Processes in Queueing Theory*. Springer, Berlin, 1976.

[Bor84] Alexandr Alekseevich Borovkov. *Asymptotic Methods in Queueing Theory*. Wiley, 1984.

[BP05] Michael Backes and Birgit Pfitzmann. Relating symbolic and cryptographic secrecy. *IEEE Transactions on Dependable and Secure Computing*, 2(2):109–123, 2005.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In *Advances in Cryptology – Eurocrypt 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155, 2000.

[Bry04] Stefan John Bryce. Factors influencing the adoption of wireless networks in New Zealand businesses. Master's thesis, University of Otago, 2004.

[BSH+98] Carlo Blundo, Alfredo De Santis, Amir Herzberg, Shay Kutten, Ugo Vaccaro, and Moti Yung. Perfectly-secure key distribution for dynamic conferences. *Information and Computation*, 146(1):1–23, 1998.

[BWZ06] Erik-Oliver Blass, Joachim Wilke, and Martina Zitterbart. A security-energy trade-off for authentic aggregation in sensor networks. In *Proceeding of the 2nd IEEE Workshop on Wireless Mesh Networks*, pages 135–137, 2006.

[CC03] Chin-Chen Chang and Chi-Yien Chung. An efficient session key generation protocol. In *Proceeding of the International Conference on Communication Technology*, pages 203–207, 2003.

[CC05] Ya-Fen Chang and Chin-Chen Chang. An efficient authentication protocol for mobile satellite communication systems. *ACM SIGOPS Operating Systems Review*, 39(1):70–84, 2005.

[CCE05] Jin-Hee Cho, Ing-Ray Chen, and Mohamed Eltoweissy. Optimization of batch rekey interval for secure group communications in wireless networks. In *Proceeding of International Conference on Wireless Networks, Communications and Mobile Computing*, volume 1, pages 522–527, 2005.

[CCE08] Jin-Hee Cho, Ing-Ray Chen, and Mohamed Eltoweissy. On optimal batch rekeying for secure group communications in wireless networks. *Wireless Networks*, 14(6):915–927, 2008.

[CCL05] Po-Jen Chuang, Tun-Hao Chao, and Bo-Yi Li. A scalable grouping random key predistribution scheme for large scale distributed sensor networks. In *Proceeding of the Third International Conference on Information Technology and Applications*, volume 2, pages 535–540, 2005.

[CCSI01] Ghassan Chaddoud, Isabelle Chrisment, Andre Schaff, and Loria Inria. Dynamic group communication security. In *Proceeding of 6th IEEE Symposium on computers and communication*, pages 49–56, 2001.

[CG06] Jan De Clercq and Guido Grillenmeier. *Microsoft Windows Security Fundamentals*. Digital Press, 2006.

[CJ03] Hung-Yu Chien and Jinn-Ke Jan. A hybrid authentication protocol for large mobile network. *The Journal of Systems and Software*, 67(2):123–130, 2003.

[CL06] Tian-Jie Cao and Dong-Dai Lin. Cryptanalysis of two password authenticated key exchange protocols based on {RSA. *IEEE communications letters*, 10(8):623–625, 2006.

[CQN02] Hao-Hua Chu, Lintian Qiao, and Klara Nahrstedt. A secure multicast protocol with copyright protection. *ACM SIGCOMM Computer Communications Review*, 32(2):42–60, 2002.

[CQZ09] Tian Jie Cao, Tao Quan, and Bo Zhang. Cryptanalysis of some client-to-client password-authenticated key exchange protocols. *Journal of Networks*, 4(4):263–270, 2009.

[Cra09] Guy Cranswick. 2009. Google apps total cost of ownership. Retrieved 12 March, 2010, from `http://www.misaustralia.com/viewer.aspx?EDP://1242704824150`.

[CS05] Yacine Challal and Hamida Seba. Group key management protocols: A novel taxonomy. *International Journal of Information Technology*, 2(1):105–118, 2005.

[Dai09] Wei Dai. Crypto++ 5.6.0 benchmarks. Technical report, 2009. Retrieved 31 October, 2009, from `http://www.cryptopp.com/benchmarks.html`.

[Dam90] Ivan Bjerre Damgard. A design principle for hash functions. In *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, volume 435 of *Lecture Notes In Computer Science*, pages 416–427, 1990.

[DGS98] Marten Van Dijk, Christian Gehrmann, and Ben Smeets. Unconditionally secure group authentication. *Designs, Codes and Cryptography*, 14(3):281–296, 1998.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[Dij95] Marten Van Dijk. A linear construction of perfect secret sharing schemes. In *Proceeding of Eurocrypt'94*, volume 950 of *Lecture Node in Computer Science*, pages 23–34, 1995.

[DPK+09] Marios D. Dikaiakos, George Pallis, Dimitrios Katsaros, Pankaj Mehra, and Athena Vakali. Cloud computing: Distributed internet computing for it and scientific research. *IEEE Internet Computing*, 12(5):10–13, September 2009.

[DR02] Joan Daemen and Vincent Rijme. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer-Verlag, 2002.

[DS81] Dorothy E. Denning and Giovanni Maria Sacco. Timestamps in key distribution protocols. *Communications of the ACM*, 24(8):533–536, 1981.

[EHG07] Ty Mey Eap, Marek Hatala, and Dragan Gasevic. Enabling user control with personal identity management. In *Proceeding of the IEEE International Conference on Services Computing SCC 2007*, pages 60–67, 2007.

[Eld09] Eric Eldon. (2009). Single sign-on service openid getting more usage. Retrieved December 19, 2009, from `http://social.venturebeat.com/2009/04/14/single-sign-on-service-openid-getting-more-usage/`.

[EP01] Jose Esteves and Joan Pastor. Enterprise resource planning systems research: An annotated bibliography. *Communications of AIS*, 7(8):2–54, 2001.

[FE09] Jean-Anne Fitzpatrick and Jennifer English. (2009). The kerberos authentication system. Retrieved December 18, 2009, from `http://www.obscure.org/~jafitz/250_p1/kerberos.htm`.

[FG96] Armando Fox and Steven D. Gribble. Security on the move: Indirect authentication using kerberos. In *Proceeding of the Second Annual International Conference on Mobile Computing and Network*, pages 155–164, 1996.

[FKC03] David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. *Role-Based Access Control*. Artech House Publishers, 2003.

[Found] OpenID Foundation. (n.d.). Openid foundation website. Retrieved 31 October, 2009, from `http://openid.net/`.

[GC98] Donald Gross and Harris M. Carl. *Fundamentals of Queueing Theory*. Wiley, 1998.

[GNY90] Li Gong, Roger Needham, and Raphael Yahalom. Reasoning about belief in cryptographic protocols. In *Proceeding of the IEEE Computer Society Symposium on Research in Security and Privacy*, pages 234–248, 1990.

[Gol07] Oded Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, 2007.

[Gon92] Li Gong. A security risk of depending on synchronized clocked. *ACM SIGOPS Operating Systems Review*, 26(1):49–53, 1992.

[Gon93] Li Gong. Increasing availability and security of an authentication service. *IEEE Journal on Selected Areas in Communications*, 11(5):657–662, 1993.

[GS03] Simson Garfinkel and Gene Spafford. *Practical Unix and Internet Security*. O'Reilly Media Inc., 3 edition, 2003.

[Haa08] Timber Haaker. *Mobile Service Innovation and Business Models*, chapter Mobile Service Bundles, pages 217–232. Springer, 2008.

[Hal94] Neil Haller. The s/key one-time password system. In *Proceedings of the Internet Society Symposium on Network and Distributed Systems*, pages 151–157, 1994.

[Han00] Ben Handley. Resource-efficient anonymous group identification. In *Proceedings of the 4th International Conference on Financial Cryptography*, volume 1962 of *Lecture Notes In Computer Science*, pages 295–312, 2000.

[Hen03] Ronda R. Henning. Vulnerability assessment in wireless networks. In *Proceeding of the Symposium on Applications and the Internet Workshops*, pages 358–362, 2003.

[HMV04] Darrel Hankerson, Alfred Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer, 2004.

[HZS03] Robert Hsieh, Zhe Guang Zhou, and Aruna Seneviratne. S-MIP: a seamless handoff architecture for mobile IP. In *Proceeding of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications INFOCOM 2003*, volume 3, pages 1774– 1784, 2003.

[Jab96] David P. Jablon. Strong password-only authenticated key exchange. *ACM SIGCOMM Computer Communication Review*, 26(5):5–26, 1996.

[JL96] William Johnston and Case Larsen. A use-condition centered approach to authenticated global capabilities: Security architectures for large-scale

distributed collaboratory environments. Technical Report 3885, Ernest Orlando Lawrence Berkeley National Laboratory, 1996.

[JLP97] Henrique Joao, Jose Legatheaux, and Jorge Paulo. A flexible object-group-oriented framework to support large scale collaborative applications. In *Proceeding of the 30th Hawaii International Conference on System Sciences*, volume Volume 1: Software Technology and Architecture, pages 82–91, 1997.

[Jou09] Antoine Joux. *Algorithmic Cryptanalysis*. Chapman & HallCRC, 2009.

[JP02] Eliane Jaulmes and Guillaume Poupard. On the security of homage group authentication protocol. In *Proceeding of the 5th International Conference Financial Cryptography*, volume 2339 of *Lecture Notes In Computer Science*, pages 106–116, 2002.

[JSB07] Xinbo Jiang, Farzad Safaei, and Paul Boustead. Enhancing the multicast performance of structured P2P overlay in supporting massively multi-player online games. In *Proceeding of 15th IEEE International Conference on Networks*, pages 124 – 129, 2007.

[JTY97] Philippe Janson, Gene Tsudik, and Moti Yung. Scalability and flexibility in authentication services: The kryptoknight approach. In *Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution*, pages 725–736, 1997.

[Kah67] David Kahn. *The Codebreakers*. Macmillan, 1967.

[Kal03] Burt Kaliski. (2003). TWIRL and RSA key size. Retrieved 31 October, 2009, from `http://www.rsa.com/rsalabs/node.asp?id=2004`.

[Kir07] Jeremy Kirk. (May 24, 2007). Researcher: RSA 1024-bit encryption not enough. *PC World, IDG News Service*, Retrieved from `http://www.pcworld.com/article/132184/researcher_rsa_1024bit_encryption_not_enough.html`.

[KKRD03] Thomas Kostas, Diane Kiwior, Gowri Rajappan, and Michel Dalal. Key management for secure multicast group communication in mobile networks. In *Proceeding of DARPA Information Survivability Conference and Exposition*, volume 2, pages 41–43, 2003.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman & Hall/CRC, 2007.

[KM02] Sharifullah Khan and Peter L. Mott. Scalable and dynamic grouping of continual queries. In *Proceedings of the Second International Conference of Advances in Information Systems*, volume 2457 of *Lecture Notes in Computer Science*, pages 31–42, 2002.

[Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[Koc96] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems. In *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, volume 1109 of *Lecture Notes in Computer Science*, 104-113, 1996.

[KPR03] Vlastimil Klma, Ondej Pokorny, and Toms Rosa. Attacking rsa-based sessions in SSL/TLS. In *Proceeding of Cryptographic Hardware and Embedded Systems*, volume 2779 of *Lecture Notes in Computer Science*, pages 426–440, 2003.

[KSW98] John Kelsey, Bruce Schneier, and David Wagner. Protocol interactions and the chosen protocol. In *Proceedings of the 5th International Workshop on Security Protocols*, volume 136 of *Lecture Notes In Computer Science*, pages 91–104. Springer-Verlag, 1998.

[Kum95] P.R Kumar. A tutorial on some new methods for performance evaluation of queueing networks. *IEEE journal on selected areas in communications*, 13(6):970–980, 1995.

[LB01] Yi Lu and Bharat Bhargava. Achieving flexibility and scalability: A new architecture for wireless network. In *Proceeding of the International Conference on Internet Computing*, pages 1105–1111, 2001.

[LB07] Men Long and Uri Blumenthal. Manageable one-time password for consumer applications. In *Proceeding of the International Conference on Consumer Electronics, Digest of Technical Papers. ICCE 2007*, pages 1–2, 2007.

[LC97] Albert Levi and M. Ufuk Caglayan. The problem of trusted third party in authentication and digital signature protocols. In *Proceeding of the Twelfth International Symposium on Computer and Information Sciences*, pages 317–324, 1997.

[LJCS08] HwanJin Lee, InKyung Jeun, Kilsoo Chun, and Junghwan Song. A new anti-phishing method in openid. In *Proceeding of The Second International Conference on: Emerging Security Information, Systems and Technologies. SECURWARE '08*, pages 243–247, 2008.

[Lop06] Javier Lopez. Unleashing public-key cryptography in wireless sensor networks. *Journal of Computer Security*, 14(5):469–482, 2006.

[Luo08] Xiao Luo. The realization of the radius security authentication. In *Proceeding of the 4th International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–4, 2008.

[LW08] Wen-Min Li and Qiao-Yan Wen. Efficient verifier-based password-authentication key exchange protocol via elliptic curves. In *Proceeding of the 2008 International Conference on Computer Science and Software Engineering*, volume 3, pages 1003–1006, 2008.

[LYGL01] Xiaozhou Steve Li, Yang Richard Yang, Mohamed G. Gouda, and Simon S. Lam. Batch rekeying for secure group communications. In *Proceedings of the 10th international conference on World Wide Web*, pages 525–534, 2001.

[MB93] Wenbo Mao and Colin Boyd. Towards formal analysis of security protocols. In *Proceeding of the Computer Security Foundation Workshop VI*, pages 147–158, 1993.

[MCR04] L.A. Martucci, T.C.M.B. Carvalho, and W.V. Ruggiero. A lightweight distributed group authentication mechanism. In *Proceeding of the 4th International Network Conference*, pages 393–400, 2004.

[Mer82] Ralph Charles Merkle. *Secrecy, Authentication and Public Key Systems*. UMI Research Press,U.S, 1982.

[MHN97] Ari Medvinsky, Matthew Hur, and B. Clifford Neuman. Public key utilizing tickets for application servers (pktapp). Technical Report RFC 2026, The Internet Engineering Task Force (IETF), http://tools.ietf.org/html/draft-ietf-cat-kerberos-pk-tapp-03, 1997. Retrieved 18 November, 2009.

[Mil01] Sandra Kay Miller. Facing the challenge of wireless security. *Computer*, 34(7):16–18, 2001.

[ML08] Jianfeng Ma and Xinghua Li. *Handbook of Research on Wireless Security*, chapter The Provably Secure Formal Methods for Authentication and Key Agreement Protocols, pages 210–235. Information Science Publishing, 2008.

[MS99] Philip MacKenzie and Ram Swaminathan. Secure network authentication with password identification. Technical report, IEEE P1363 working group, 1999.

[MSS98] John C. Mitchell, Vitaly Shmatikov, and Ulrich Stern. Finite-state analysis of SSL 3.0. In *Proceedings of the 7th conference on USENIX Security Symposium*, volume 7, pages 201–206, 1998.

[MvOV96] Alfred Menezes, Paul van Oorschot, and Scott Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[NKW04] Junghyun Nam, Seungjoo Kim, and Dongho Won. Attacks on bresson-chevassut-essiari-pointcheval's group key agreement scheme for low-power mobile devices. Technical Report Report 2004/251, Cryptology ePrint Archive, 2004.

[NS78] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

[NT94] B. Clifford Neuman and Theodore Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications*, 32(9):33–38, 1994.

[NWL+09] Huy Hoang Ngo, Yiling Wang, Phu Dung Le, Balasubramaniam Srinivasan, and Vishv Malhotra. A membership oriented group key management for application services. In *Proceeding of the 12th International Conference on Network-Based Information Systems*, 2009.

[NWL+10] Huy Hoang Ngo, Xianping Wu, Phu Dung Le, Campbell Wilson, and Balasubramaniam Srinivasan. Dynamic key cryptography and applications. *International Journal of Network Security*, 10(3):161–174, 2010.

[OJ08] Hyun-Kyung Oh and Seung-Hun Jin. The security limitations of sso in openid. In *The 10th International Conference on Advanced Communication Technology, ICACT 2008.*, volume 3, pages 1608–1611, 2008.

[Opp96] R. Oppliger. *Authentication Systems for Secure Networks*. Artech House, 1996.

[Opp03] Rolf Oppliger. Microsoft .net passport: A security analysis. *Computer*, 36(7), 2003.

[PACB98] R. Poovendram, S. Ahmed, S. Corson, and J. Baras. A scalable extension of group key management protocol. In *Proceedings of the 2nd Annual ATRIP Conference*, pages 187–191, 1998.

[Par10] Ben Parr. 2010. Google wave: A complete guide. Retrieved 12 March, 2010, from `http://www.google.com.au/apps/`.

[PDM03] Roberto Di Pietro, Antonio Durante, and Luigi V. Mancini. A reliable key authentication schema for secure multicast communications. In *Proceeding of the 22nd International Symposium on Reliable Distributed Systems (SRDS'03)*, pages 231–241, 2003.

[PGW06] Raphael C. W. Phan, Bok-Min Goi, and Kah-Hoong Wong. Cryptanalysis of some improved password-authenticated key exchange schemes. *Computer communications*, 29(15):2822–2829, 2006.

[PJH08] Soohong Park, Seong-Ho Jeong, and Cheolju Hwang. Mobile IPTV expanding the value of IPTV. In *Proceeding of the Seventh International Conference on Networking*, pages 296–301, 2008.

[PM04] Asad Amir Pirzada and Chris McDonald. Kerberos assisted authentication in mobile ad-hoc networks. In *Proceedings of the 27th Australasian Computer Science Conference*, pages 41–46, 2004.

[PP03] Souradyuti Paul and Bart Preneel. Analysis of non-fortuitous predictive states of the RC4 keystream generator. In *Proceeding of INDOCRYPT*, pages 52–67, 2003.

[PQ02] Olivier Pereira and Jean-Jacques Quisquater. Some attacks upon authenticated group key agreement protocols. *Journal of Computer Security*, 11(4):555–580, 2002.

[PWFK02] Laura Pearlman, Von Welch, Ian Foster, and Carl Kesselman. A community authorization service for group collaboration. In *Proceeding of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks*, pages 50–59, 2002.

[Ray03] Eric S. Raymond. *The Art of UNIX Programming*. Addison-Wesley Professional, 2003.

[RBK08] M. De Reuver, Harry Bouwman, and T.De Koning. *Mobile Service Innovation and Business Models*, chapter The Mobile Context Explored, pages 89–114. Springer, 2008.

[RH93] Aviel D. Rubin and Peter Honeyman. Formal methods for the analysis of authentication protocols. Technical Report CITI Technical Report 93-7, Center for Information Technology Integration, 1993.

[RH03] Sandro Rafaeli and David Hutchinson. A survey of key management for secure group communication. *ACM Computing Surveys*, 35(3):309–329, 2003.

[RR09] John Rittinghouse and James Ransome. *Cloud Computing: Implementation, Management, and Security*. CRC Press, 2009.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard Max Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[RSA06] RSA. 2006. Wireless adoption increases, security improves in worlds major cities. Retrieved 12 January, 2010, from `http://www.rsa.com/press_release.aspx?id=6870`.

[Sal04] Apostolis K. Salkintzis. *Mobile Internet Enabling Technologies and Services*. CRC press LLC, 2004.

[SC01] Paul Syverson and Iliano Cervesato. The logic of authentication protocols. In *Processing of Foundations of Security Analysis and Design*, volume 2171 of *Lecture Notes In Computer Science*, pages 63–136, 2001.

[SCFY96] Ravi Sandhu, Edward Coyne, Hal Feinstein, and Charles Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[Sch96] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[Sch05] Bruce Schneier. Two-factor authentication: Too little, too late. *Communications of The ACM*, 48(4):136, 2005.

[Sha49] Claude Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):657–715, 1949.

[Shi07] Kyung-Ah Shim. Potential weaknesses of autha password-authenticated key agreement protocols. *Computer Standards & Interface*, 29:580–583, 2007.

[SHK06] Takamichi Saito, Ryosuke Hatsugai, and Toshiyuki Kito. On compromising password-based authentication over HTTPS. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications*, volume 1, pages 869–874, 2006.

[SN03] Chetan Sharma and Yasuhisa Nakamura. *Wireless Data Services Technologies, Business Models and Global Markets*. Cambridge University Press, 2003.

[SNS88] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems (version 4). In *Proceedings of the Winter 1988 Usenix Conference*, pages 191–202, 1988.

[SO94] Paul Syverson and Paul C. Van Oorschot. On unifying some cryptographic protocol logics. In *Proceedings of the 1994 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 14–28, 1994.

[Sti05] Douglas R. Stinson. *Cryptography: Theory and Practice*. Chapman & Hall/CRC, 3 edition edition, 2005.

[STL04] Yan Sun, Wade Trappe, and K. J. Ray Liu. A scalable multicast key management scheme for heterogeneous wireless networks. *IEEE/ACM Transactions on Networking*, 12(4):653–666, 2004.

[SYS97] Shiuh-Pyng Shieh, Wen-Her Yang, and Hun-Min Sun. An authentication protocol without trusted third party. *IEEE Communications Letters*, 1(3), 1997.

[Syv93] Paul Syverson. On key distribution protocols for repeated authentication. *ACM SIGOPS Operating Systems Review*, 27(4):24–30, 1993.

[Syv94] Paul Syverson. A taxonomy of replay attacks. In *Proceedings of the 7th IEEE Computer Security Foundations Workshop*, pages 187–191, 1994.

[Tar03] Peter Tarasewich. Designing mobile commerce applications. *Communications of the ACM*, 46(12):57–60, 2003.

[TLP03] Wei Tang, Ling Liu, and Calton Pu. Trigger grouping: A scalable approach to large scale information monitoring. In *Proceeding of the Second IEEE International Symposium on Network Computing and Applications*, pages 148–155, 2003.

[TNH+00] Brian Tung, Clifford Neuman, Matthew Hur, Ari Medvinsky, Sasha Medvinsky, John Wray, and Jonathan Trostle. Public key cryptography for initial authentication in kerberos. Technical Report RFC 1510, The Internet Engineering Task Force, http://zinfandel.levkowetz.com/html/draft-ietf-cat-kerberos-pk-init-11, 2000. Retrieved 18 November, 2009.

[TRN+98] Brian Tung, Tatyana Ryutov, Clifford Neuman, Gene Tsudik, Bill Sommerfeld, Ari Medvinsky, and Matthew Hur. Public key cryptography for cross-realm authentication in kerberos. Technical Report RFC 1510, The Internet Engineering Task Force (IETF), http://tools.ietf.org/html/draft-ietf-cat-kerberos-pk-cross-04, 1998. Retrieved 18 November, 2009.

[TvS06] Andrew S. Tanenbaum and Maarten van Steen. *Distributed Systems: Principles and Paradigms*. Prentice Hall, 2006.

[TW06] John Talbot and Dominic Welsh. *Complexity and Cryptography: An Introduction*. Cambridge University Press, 2006.

[UD06] Hwayoung Um and Edward J. Delp. A secure group key management scheme for wireless cellular networks. In *Proceeding of the Third International Conference on Information Technology: New Generations (ITNG'06)*, pages 414–419, 2006.

[Var05] Vijay Varadharajan. Authorization and trust enhanced security for distributed applications. In *Proceeding of the First International Conference in Information Systems Security, ICISS 2005*, Lecture Notes in Computer Science, pages 1–20, 2005.

[VMC02] John Viega, Matt Messier, and Pravir Chandra. *Network security with OPENSSL*. O'Reilly, 2002.

[vO93] Paul C. van Oorschot. Extending cryptographic logics of belief to key agreement protocols. In *Proceeding of the 1st ACM Conference on Computer and Communications Security*, pages 232–243, 1993.

[VRMCL09] Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: Towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55, 2009.

[Wan09] Yiling Wang. *Key Management for Secure Group Applications in Wireless Networks*. PhD thesis, Monash University, 2009.

[WGE+05] Arvinderpal S. Wander, Nils Gura, Hans Eberle, Vipul Gupta, and Sheueling Chang Shantz. Energy analysis of public-key cryptography for wireless sensor networks. In *Proceeding of the Third IEEE International Conference Pervasive Computing and Communications*, pages 324–328, 2005.

[WGL98] Chung Kei Wong, Mohamed Gouda, and Simon S. Lam. Secure group communications using key graphs. *ACM SIGCOMM Computer Communication Review*, 28(4):68–79, 1998.

[Wiknd] Wikipedia. (n.d.). Web service. Retrieved 31 October, 2009, from `http://en.wikipedia.org/wiki/Web_service`.

[WL92] Thomas Y.C. Woo and Simon S. Lam. Authentication for distributed systems. *Computer*, 25(1):39–52, 1992.

[WLS07] Yiling Wang, Phu Dung Le, and Balasubramaniam Srinivasan. Hybrid group key management scheme for secure wireless multicast. In *Proceeding of the 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007)*, pages 346–351, 2007.

[Wor09] OAuth Core Workgroup. (2009). Oauth core 1.0 revision a. Retrieved 31 October, 2009, from `http://oauth.net/core/1.0a`.

[Wu97] Thomas Wu. The secure remote password protocol. In *Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, pages 97–111, 1997.

[Wu98] Thomas Wu. The secure remote password protocol. In *Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, pages 97–111, 1998.

[Wu09] Xianping Wu. *Security Architecture for Sensitive Information Systems*. PhD thesis, Monash University, 2009.

[WW04] Zhiguo Wan and Shuhong Wang. Cryptanalysis of two password-authenticated key exchange protocols. In *Proceeding of the Australasian Conference on Information Security and Privacy*, volume 3108 of *Lecture Notes in Computer Science*, pages 164–175, 2004.

[WZ08] Shuhua Wu and Yuefei Zhu. Password-authenticated key exchange between clients in a cross-realm setting. In *Proceeding of IFIP International Conference in Network and Parallel Computing*, volume 5245 of *Lecture Notes in Computer Science*, pages 94–104, 2008.

[YS03] Su Jung Yu and Joo-Seok Song. An improved password authentication key exchange protocol for 802.11 environment. In *International Conference in Computational Science and Its Applications*, pages 201–209, 2003.

[YY05] Eun-Jun Yoon and Kee-Young Yoo. Cryptanalysis of password authenticated key exchange based on RSA for imbalanced wireless networks. *IEICE transactions on communications*, 88(6):2627–2628, 2005.

[YYC05] Chou-Chan Yang, Ya-Wen Yang, and Ting-Yi Chang. Cryptanalysis of an authentication key exchange protocol. *Journal of Applied Sciences*, 5(2):281–283, 2005.

[ZCY+08] Yang Zhang, Yifan Chen, Rong Yu, Supeng Leng, Huansheng Ning, and Tao Jiang. *Handbook of Research on Wireless Security*, chapter XXI: Access

Security in UMTS and IMS, pages 339–350. Information Science Reference, 2008.

[ZLR09] Xiaodong Zuo, Fengmei Liu, and Chuanlun Ren. Cryptanalysis of a password-based key exchange protocol. In *Proceeding of the Second International Conference on Innovative Computing, Information and Control*, pages 610–613, 2009.

[ZMBL04] Jun Zhang, Fanyuan Ma, Yingcai Bai, and Minglu Li. Performance analysis of batch rekey algorithm for secure group communications. In *Proceeding of the 5th International Conference of Parallel and Distributed Computing: Applications and Technologies*, volume 3320 of *Lecture Notes in Computer Science*, pages 829–832, 2004.

[ZPS93] Yuliang Zheng, Josef Pieprzyk, and Jennifer Seberry. Haval - a one-way hashing algorithm with variable length of output. In *Proceeding of International Conference on Cryptology: Advances in Cryptology, AUSCRYPT*, volume 718, pages 81–104. Springer-Verlag, 1993.

[ZRM05] Xukai Zou, Byrav Ramamurthy, and Spyros S. Magliveras. *Secure Group Communications Over Data Networks*. Springer, 2005.

[Zur96] Mary Ellen Zurko. 1996. Adage: Authorization for distributed applications and groups. Retrieved 8 February, 2010, from `http://csrc.nist.gov/nissc/1996/papers/NISSC96/lunt/ZURKO.PDF`.