Malaysian Journal of Computer Science, Vol. 12 No. 1, June 1999, pp. 60-66

DO WE NEED DIFFERENT PROTOCOLS TO SUPPORT GROUP-ORIENTED CRYPTOSYSTEM?

L. A. Gumel
Dept. of Mathematics
Universiti Putra Malaysia
43400 UPM, Serdang
email: gs00607@stud.upm.edu.my

Soo Kar Leow
Dept. of Mathematics
Universiti Putra Malaysia
43400 UPM, Serdang

A. K. Ramani
Dept. of Computer Science
Universiti Putra Malaysia
43400 UPM, Serdang

ABSTRACT

Introduces some of the commonly known methods (protocols) used in key sharing/distribution problems, and finally shows that the existing protocols can be extended to support Group-Oriented Cryptoscheme (GOC). It is both a tutorial and an introduction to GOC.

Keywords: ***Network protocols, Network security, Secret sharing***

1.0 INTRODUCTION

Computer networks are receiving a great attention today as a means of improving human to computer and computer to computer communications. The world is moving from the industrial age to the golden age of information. In this juncture, communication systems are considered to be the backbone of the system. Unfortunately, every computer system and network that transmit and store readable information is vulnerable to attack by an intruder (enemy). Hence, valuable information of any kind needs to be protected against unauthorized access or alteration.

Historically, cryptography has been used long before the invention of computers to secure sensitive military and diplomatic communications. However, with the introduction of conventional and public key cryptosystems, cryptology is now widely used to provide data security. According to Cambell [1], cryptography can be used to provide three aspects of security: data security, data authentication, and originator authentication.

2.0 NETWORK SECURITY

The goal of network security is to ensure the availability of information and information processing resources, and provide means to establish and retain the integrity and confidentiality of information within the system. According to Daniel [2], the types of attacks on the security of any computer system or network can be classified into:

- *Interruption* (An attack on availability)
- *Interception* (Attack on confidentiality)
- *Modification* (Attack on integrity)
- *Fabrication* (Attack on authentication).

The above mentioned attacks can be categorized in terms of *passive attack* (such as eavesdropping, traffic analysis and wiretapping), or *active attack* (such as masquerade, replay and denial of service) [3]. Some of the possible security threats are listed below:

- *Traffic analysis*- the observation of information about a communication between users. The observation may include the absence or presence of traffic frequency, direction, sequence, type and amount of traffic.

- *Replay*- the recording and subsequent replay of communication at some later point in time.
- *Identity interception*- the observation of the identity of one or more parties involved in a communication for misuse.

- *Masquerading*- the impersonation of a user to gain access to information, or to gain accidental privileges.

- *Mis-routing*- the misrouting of a communication path intended for one user to another.

- *Unauthorized access*- the unauthorized usage of resources and access to classified data by an intruder.

3.0 AUTHENTICATION PROTOCOLS AND KEY EXCHANGE

The main goal of authentication is to make two entities believe that they are communicating with each other and not with intruders. Authentication can be classified into two categories:

1. *Simple authentication*: A case where by only the name and the password supplied by the sender are checked by the receiver.

2. *Strong authentication*: A case in which cryptographic techniques are used to protect the exchange of validating information.

All classical encryption methods suffer from the key distribution problems. For conventional encryption, the storage and communication of the keys are the most important measures of the security. Likewise, in the case of public key cryptosystems, the secure transmission of keys to the users who need them was identified to be a major problem [4]. Simmons provides more details about the two schemes [5].

Some of the well-known protocols for authentication and key distribution problems are given below.

3.1 Needham-Schroedar Protocol

The protocol was invented by Needham and Schroedar [6]. It uses symmetric encryption and a trusted third party called an authenticator server **S** which holds the secret key of all communicating entities. The protocol is shown in Fig. 1. The protocol may be summarized with the following message exchange:
Message 1: A sends its identity and the identity of B, together with a nonce (Na) to S A⇒S: (A[B, Na])
Message 2: S generates a session key K, and communicates it secretly to both A and B, by encrypting K and the initiator A with the secret key of B(Kb), and again encrypts the same together with Na and K with A secret key (Ka) and sends it to A. S⇒A: {(Na, B, K, [K,A] Kb) Ka}
Message 3. A sends the part of the message encrypted with Kb to B. A⇒B: ({ [K,A]}Kb)
Message 4. B sends to A a nonce Nb encrypted with K, to confirm that it is not a replay. B⇒A: ({Nb}K)
Message 5. A decrypts the message with K, generates [Nb-1] and encrypts it with K. Then sends it back to B. A⇒B: ({Nb-1}K)
Message 6. B decrypts the message with K and verifies that it is not a replay. Na, Nb and Nb-1 is to guarantee that there are no replay attacks.

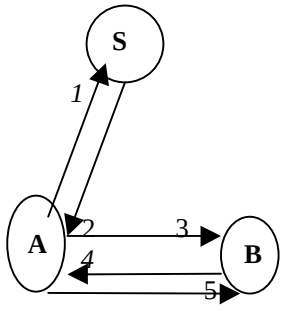

Fig. 1: Needham-Schroedar Protocol

3.2 CCITT X.509 Protocol

Unlike the Needham-Schroedar protocol, the CCITT X.509 protocol [7] is based on asymmetric encryption technique. The framework is certificate-based. The credentials of users are stored as certificates, which are signed by a common trusted third party known as a Certification Authority (CA). If A and B want to communicate, each has to verify the signature of the other person's certificate.

Unlike the Needham-Schroedar protocol, the CCITT X.509 protocol [7] is based on asymmetric encryption technique. The framework is certificate-based. The credentials of users are stored as certificates, which are signed by a common trusted third party known as a Certification Authority (CA). If A and B want to communicate, each has to verify the signature of the other person's certificate.

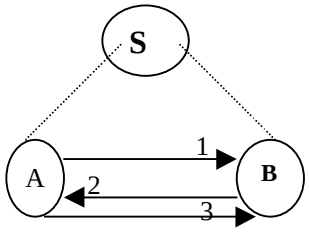

Fig. 2: X.509 Protocol

To begin with the Protocol as shown in Fig. 2. A generates a nonce Na and a confidential data Ya, and transmits the following message to B:
Message 1. A⇒B: [A,{Na, B, Xa, (Ya) Kb}Ka $^{-1}$]
where Ya could be some data to be transferred or could be a session key for subsequent exchange of data. Xa is some data whose integrity needs to be maintained. On receipt of message 1, B obtains the public key of A and verifies that A's certificate has not expired, signature of A (integrity), and freshness of Na, and then sends the following message:
Message 2. B⇒A: [B{Nb, A, Na, Xb, (Yb) Ka}Kb $^{-1}$]
on receipt of message 2 A also performs similar verifications as above, and sends the message below to B:
Message 3. A⇒B: [A{Nb}Ka $^{-1}$]
At the end of the three messages (1,2,3), both parties have authenticated each other. Hence A and B may build a common secret based on Ya and Yb which can be a shared key say Kab = f(Ya,Yb).

3.3 Kerberos Protocol

Kerberos is a trusted third-party authentication protocol based on the symmetric technique. It is probably the most widely used authentication service today. The Kerberos [8] protocol establishes a shared key between two entities wishing to communicate with the help of the authentication server. When a user requests a service, his identity must be established. This is done by presenting a ticket to the server along with a proof that the ticket was originally issued to the user and is not a replay. There are two types

of credentials that are used in the Kerberos protocol. One is ticket, and the other one is authenticator, as shown in Fig. 3. A Kerberos protocol contains the key material (session key) to be shared by the client and server (entities). The trusted third-party (KS) generates unique, fresh and good quality strong keys. The protocol can be summarized as follows:
Message 1. The client C requests a ticket from Kerberos (KS): C ⇒ KS: {C, TGS}.

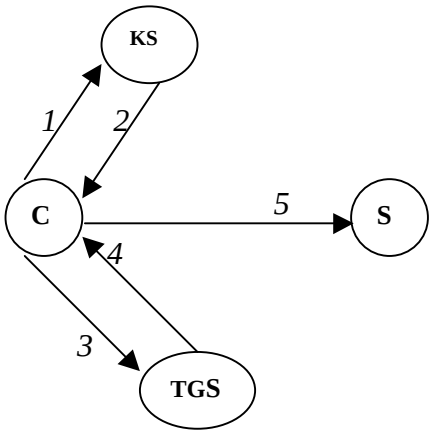

Fig. 3: Kerberos protocol

Kerberos transmits the ticket for TGS (ticket granting servers) to C, which includes a session key $K_{c,tgs}$ to be used between C and TGS, time-stamp and life-time for the ticket encrypted with K_{tgs} and a copy of the session key. This is encrypted with the master key of user C, which is stored by Kerberos, as shown below:
Message 2. KS ⇒ C: $[\{K_{c,tgs}, (T_{c,tgs}) K_{tgs}\}K_c]$
Now the user C requests authentication by generating an authenticator A_c and presents it together with the ticket obtained from Kerberos to the TGS requesting another ticket to the actual server S:
Message 3. C ⇒ TGS: $[S,\{A_c\}K_{c,tgs},\{ T_{c,tgs}\}K_{tgs}]$
TGS will then generate the ticket after verifying the validity of the ticket and the authenticator as follows:
Message 4. TGS ⇒ C: $[\{ (T_{c,s})K_{s,} K_{c,s} \} K_{c,tgs}]$
Finally the user presents the ticket obtained from TGS to the actual end server S:
Message 5. C ⇒ S: $[\{ A_c\}K_{c,s,} (T_{c,s}) K_s]$

In a networking environment, a secret session key needs to be securely communicated between users prior to communications. As the number of users in the network become larger, key distribution and management will become a serious problem. All the above mentioned protocols have a common drawback. They are dealing with communication between two users and thus key management becomes a major problem. For that reason, we proposed the new scheme below as a means of reducing the number of keys needed for secure communication.

4.0 THE PROPOSED SCHEME

There is no doubt that secured communications can be achieved if a common key distribution is obtainable between entities. It also offers an advantage for key management in large open networks. The proposed method is based on the Needham-Schroeder protocol and a symmetrical square matrix. S is regarded as the *mutually trusted authority* (MTA) or the *dealer* (D) for all the groups and it shares a key with each group. The key is generated by secret sharing. Since S is the dealer, it has an identity of each group so that it can develop a matrix M and send it to each of the communicating groups to establish the session key.

Secret sharing schemes are used in information security theory, like modeling access control and cryptographic key distribution problems [9]. It is a method of sharing a secret key among a set of users (participants) $\boldsymbol{U}$ in such a way that only certain specified users are qualified to compute the secret key by combining their shares. This is done by the MTA or a special participant called the dealer.

Let $\boldsymbol{U} = \{Ui: 1 \leq i \leq n\}$ be the set of users, we assume that MTA $\notin \boldsymbol{U}$, $\exists$ a secret key $K \in K$, K is the key set (i.e., the set of all possible keys), S is already defined above, let $\boldsymbol{\Gamma}$ be a set of subsets of $\boldsymbol{U}$, this is denoted mathematically by the notation $\boldsymbol{\Gamma} \subseteq 2^U$. The subsets in $\boldsymbol{\Gamma}$ are those subsets of users that should be able to compute the secret. $\boldsymbol{\Gamma}$ is called an access structure and the subsets in $\boldsymbol{\Gamma}$ are called authorized subsets while any subsets in $\boldsymbol{\Gamma}^c = \{X/X \notin \boldsymbol{\Gamma}\}$ are all forbidden or unauthorized subsets, the MTA or S selects a secret s among a set of possible secrets S according to a probability distribution on S, let F be a set of distribution rules denote

$$F\mathrm{K} = \{ f \in F : f(\mathrm{D}) = \mathrm{K}\}.$$

Then the MTA selects the share for each of the legal users by means of secret sharing scheme. The scheme is said to be a *perfect secret-sharing scheme* if the following two properties are satisfied:

1) If α is a legal subset of $\boldsymbol{U}$ (positive access instances), i.e., if $\alpha \subseteq \boldsymbol{U}$ pool their shares, then they can determine the value of K.
2) If β is not a legal subset of $\boldsymbol{U}$ (negative access instances), i.e., if $\beta \subseteq \boldsymbol{U}$ pool their shares, then they cannot determine the value of K.

The above properties can be achieved if the following holds:

i) Let $\alpha \subseteq \boldsymbol{\Gamma}$ and $f, g \in F$. If $f(Ui) = g(Ui)\ \forall\ Ui \in \alpha$, then $f(\mathrm{D}) = g(\mathrm{D})$.

ii) Let $\beta \subseteq \boldsymbol{\Gamma}^c$ and $f : \beta \rightarrow S$. Then $\exists$ a non-negative integer $\lambda(f, \beta)$ such that, $\forall\ \mathrm{K} \in K$, $|\{ g \in F\mathrm{K} : g(Ui) = f(Ui)\ \forall\ Ui \in \beta\}| = \lambda(f, \beta)$

4.1 How M is Generated

In this proposed scheme, two groups of entities (say group1 and group 2) can authenticate each other in the following manner: Suppose S generates a symmetrical matrix M (where the elements of M are numeric) and keeps it secret. Let group1 and group2 (G_1 and G_2) be in the system, G_1 sends its ID vector $IG_1 = (I\delta_1, I\delta_2, \ldots\ldots I\delta_n)$ to the center, then the secret matrix is computed as $SG_1 = IG_1.M$.

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & \ldots\ldots & m_{1n} \\ m_{12} & m_{22} & m_{23} & \ldots\ldots & m_{2n} \\ m_{13} & m_{23} & m_{33} & \ldots\ldots & m_{3n} \\ m_{14} & m_{24} & m_{34} & \ldots\ldots & m_{4n} \\ . & . & . & \ldots\ldots & m_{5n} \\ . & . & . & \ldots\ldots & . \\ . & . & . & \ldots\ldots & . \\ m_{1n} & m_{2n} & m_{3n} & \ldots\ldots & m_{nn} \end{bmatrix}$$

Now, the center computes G_1 secret information $SG_1 = IG_1.M$ as follows:

$$(I\delta_1, I\delta_2, \ldots\ldots, I\delta_n) \begin{bmatrix} m_{11} & m_{12} & m_{13} & \ldots\ldots & m_{1n} \\ m_{12} & m_{22} & m_{23} & \ldots\ldots & m_{2n} \\ m_{13} & m_{23} & m_{33} & \ldots\ldots & m_{3n} \\ m_{14} & m_{24} & m_{34} & \ldots\ldots & m_{4n} \\ . & . & . & \ldots\ldots & m_{5n} \\ . & . & . & \ldots\ldots & \\ . & . & . & \ldots\ldots & \\ m_{1n} & m_{2n} & m_{3n} & \ldots\ldots & m_{nn} \end{bmatrix}$$

Group1 can then compute the common key using IG_2
$KG_1G_2 = SG_1.IG_2^T = IG_1.M.IG_2^T$
Group2 can similarly compute the common key KG_2G_1 using IG_1
ie $KG_2G_1 = SG_2.IG_1^T = IG_2.M.IG_1^T$
Hence both group1 and group2 obtained a common key $KG_1G_2 = KG_2G_1$

To prove that $KG_1G_2 = KG_2G_1$ (i.e., both groups have a common key), we look at the commutativity of the matrix.
$KG_1G_2 = G_1.M.IG_2^T$

$$= (I\delta_1, I\delta_2, \ldots\ldots I\delta_n) \begin{bmatrix} m_{11} & m_{12} & m_{13} & \ldots & m_{1n} \\ m_{12} & m_{22} & m_{23} & \ldots\ldots & m_{2n} \\ & & \ldots\ldots & & \\ m_{14} & m_{24} & m_{34} & \ldots\ldots & m_{4n} \\ & & \ldots\ldots & & \\ & & \ldots\ldots & & \\ & & & & \\ m_{1n} & m_{2n} & m_{3n} & \ldots\ldots & m_{nn} \end{bmatrix} \begin{bmatrix} I\chi_1 \\ I\chi_2 \\ I\chi_3 \\ I\chi_4 \\ . \\ . \\ . \\ I\chi_n \end{bmatrix}$$

$= (I\delta_1 m_{11} + I\delta_2 m_{12} + \ldots + I\delta_n m_{1n})\, I\chi_1 + (I\delta_1 m_{12} + I\delta_2 m_{22} + \ldots + I\delta_n m_{2n})\, I\chi_2 + \ldots + (I\delta_1 m_{1n} + \ldots\ldots\ldots\ldots + I\delta_n m_{nn})\, I\chi_n$

$= I\delta_1 m_{11} I\chi_1 + I\delta_2 m_{12} I\chi_1 + \ldots + I\delta_n m_{1n} I\chi_1 + I\delta_1 m_{12} I\chi_2 + \ldots + I\delta_n m_{2n} I\chi_2 + \ldots\ldots +$
$I\delta_1 m_{1n} I\chi_n + \ldots\ldots + I\delta_n m_{nn} I\chi_n$

Rearranging the terms we get:

$KG_1G_2 = (I\chi_1 m_{11} + I\chi_2 m_{12} + \ldots\ldots + I\chi_n m_{1n})\, I\delta_1 + (I\chi_1 m_{12} + I\chi_2 m_{22} + \ldots\ldots + I\chi_n m_{2n})\, I\delta_2 + \ldots\ldots + (I\chi_1 m_{1n} + I\chi_2 m_{2n} + \ldots\ldots + I\chi_n m_{nn})\, I\delta n$

$$= \{(I\chi_1 m_{11} + I\chi_2 m_{12} + \ldots\ldots + I\chi_n m_{1n}), (I\chi_1 m_{12} + I\chi_2 m_{22} + \ldots\ldots + I\chi_n m_{2n}), \ldots, (I\chi_1 m_{1n} + I\chi_2 m_{2n} + \ldots\ldots + I\chi_n m_{nn})\} \cdot \begin{bmatrix} I\delta_1 \\ I\delta_2 \\ . \\ . \\ . \\ . \\ I\delta n \end{bmatrix}$$

$$= (I\chi_1, I\chi_2, \ldots, I\chi_n) \begin{bmatrix} m_{11} & m_{12} & m_{13} & \ldots & m_{1n} \\ m_{12} & m_{22} & m_{23} & \ldots & m_{2n} \\ m_{13} & m_{23} & m_{33} & \ldots & m_{3n} \\ m_{14} & m_{24} & m_{34} & \ldots & m_{4n} \\ . & . & \ldots & m_{5n} & \\ . & . & . & . & \ldots \\ . & . & . & . & \ldots \\ m_{1n} & m_{2n} & m_{3n} & \ldots & m_{nn} \end{bmatrix} \begin{bmatrix} I\delta_1 \\ I\delta_2 \\ I\delta_3 \\ . \\ . \\ . \\ . \\ I\delta_n \end{bmatrix}$$

$= IG_2.M.IG_1^T$

Therefore, it is proved that $KG_1G_2 = KG_2G_1$, and hence both the groups shared the same key. It should be noted that this key is only a session key between the groups. The ID of each group should be constructed in such a way that it consists of some information with regard to each member in each group, hence it is easy to deal with a drop member by notifying the center. Moreover, the lifetime of each session key should be a short period (not exceeding one day). It is clear that a drop member alone cannot reconstruct the secret, since if m shadows are needed to reconstruct the secret, then even m-1 of the shadows cannot reconstruct it. For security and efficiency, it is advisable to keep the size of the shares as small as possible.

Since we are dealing with entire group rather than individuals in the group, any message here is considered to be an *important message* (i.e., only set of legal recipients can decipher it). But if any member is considered to be more important than others, he can be given more than one share or even the minimum number of shares needed to decipher the secret if he alone is permitted to decipher the secret. It should be noted that the security of the whole system depends on the matrix. Therefore, it should be kept secret and different matrices should be computed for different session keys.

Another important aspect in network security is how to make sure that a message received is not a replay. There are basically three commonly known techniques: *Challenge-response nonces, timestamps, and sequence numbers*. In this scheme, we propose the use of sequence numbers between the groups instead of nonces or timestamps. A sequence number protocol has the advantage of reducing the delays before secure communication starts. It also has an important feature that each individual message carries sufficient information to enable its recipients to verify the freshness of the message. The only well-known (most serious) disadvantage of sequence numbers is that maintaining separate sequence numbers for each communicating entity in a large network is not usually practicable, since our approach is purposely to take care of such problem, hence the technique will be quite suitable.

Challenge-response has some drawbacks that will make it unsuitable in our scheme. Unlike sequence number, it requires a party (or a group in this case) to sign a random number chosen by another group [10]. Timestamp is not flexible enough to be a general security mechanism in distributed system [11], as it assumes the presence of a globally accessible clock. It should be noted that message delivery has a finite speed and that distributed clocks are unlikely to have identical values at all time.

The proposed scheme can be summarized as follows:

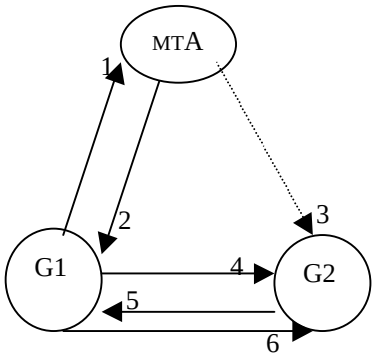

Fig. 4: Summary

Steps:
Message 1: G1 ⇒ MTA: (IG1[IG2,N1])
Message 2: MTA⇒G1: {(N1,IG2,SG1[SG2,G1]KG2)KG1}
Message 3: MTA⇒G2: ([SG2,G1]KG2)
Message 4: G1⇒G2: ([SG2,G1]KG2)
Message 5: G2⇒G1: ([Sng2]KG2G1)
Message 6: G1⇒G2: ([Sng2-1]KG1G2)
The notations mean:
IG1→ Identity of group 1
IG2→ Identity of group 2
N1→ Nonce of group 1
SG1→ group 1 secret information
SG2→ group 2 secret information
KG1→ secret key shared between MTA and group 1
KG2→ secret key shared between MTA and group 2
Sng2→ Sequence number generated by group 2

Note that message 3 is confirming message 4 to group 2. Messages 5 and 6 are proving that the session keys computed by both groups are the same.

5.0 BRIEF COMPARISON TO OTHER METHODS

Some schemes have been proposed to implement the group oriented cryptosystems (GOC). Desmedt [13], first proposed a method to solve GOC problems, but his method proved to be of theoretical interest only. Due to such drawback, a more practical solution to the GOC problems was proposed by Frankel [14], that presented a protocol which partially solves the GOC problem, the bottleneck of this scheme is that it requires two trusted clerks in each group. In the sending group, these two clerks are responsible for sending the encrypted messages to the destination group, while in the receiving group, these two clerks are then responsible for distributing the received messages to the recipients according to the security policy of the group. Thus, the entities also have to trust some "institution" behind the system. From the point of view of protocol efficiency, this approach is very inefficient. Desmedt and Frankel scheme [15] uses a fixed secret key whose shares should be securely distributed to a given group with predetermined threshold parameters when the system is set up. One of the drawbacks of the scheme is that there may be a collusion of members to discover the group secret.

Our approach is quite different, it possesses some advantages compared to the others. For instance, any group can change its ID vector for security reason without affecting others (before establishing a particular session key with any group). Furthermore, some entities can easily be inserted or cancelled from the system. It is also possible for a single recipient to decipher the encrypted message (urgent messages only) or if he possesses the minimum number of shares. Moreover, the users in each group need not authenticate other users from different groups as long as they have been authenticated by the MTA. It automatically detects any attempt to replay a message by means of sequence number.

6.0 ADVANTAGES AND DISADVANTAGES OF GROUP-ORIENTED CRYPROSYSTEMS

In a user-to-user system, public keys or secret keys can be used to establish a session key. However, as the size of network increases, the quantity of keys increases to the point where key management becomes a major problem. In general, some of the advantages and disadvantages of GOCs over two entities cryptosystems are as follows:

Advantages

- It reduces the problem of providing protection for a large number of keys to that of protecting a small number of keys
- Unlike in peer-to-peer system, exposure of session keys is almost impossible
- Achieve maximum flexibility with regard to specific key distribution protocols and specific cryptographic algorithms
- It can easily generate or acquire and distribute keys to group of entities instead of peer-to-pear (user-to-user).

Disadvantages

- The MTAs must be trusted by all nodes on the network, so it is a convenient central point of attack
- It is too difficult and expensive to implement, and potentially too costly for many users and organizations
- There are significant potential risks such as failure, especially when using a single MTA.

7.0 CONCLUSION

Network security is a collection of services which provide and maintain the authenticity, integrity, confidentiality and availability of data, and to make sure of non-repudiation.

We have seen that a new scheme is needed if group of entities authentication and key distribution want to be considered in secure communications. Almost all the current known protocols are designed for communications between two parties, which is somewhat infeasible in large networks. The proposed method takes advantage of MTA to distribute part of the shared key to be used between groups, which will be securely communicated by means of secret sharing techniques and Needham-Schroeder protocol. One of the significant results achieved lies in the ability to detect replay of messages and at the same time allows a message from a given group to be authenticated by another group.

The work carried out here provides an impetus needed for further research to test out different strategies, concepts and methodologies in the area of Group Oriented Cryptosystems.

REFERENCES

[1] C. M. Campbell, "Design and Specification of Cryptographic Capabilities ", *IEEE Communications Magazine,* Vol. 16, No. 6, pp. 15-19. Nov. 1978.

[2] Daniel et al., "Secure Communications in ATM Networks", *Communications of the ACM,* Vol. 38, No. 2, pp. 45-52, 1995.

[3] ISO Information Processing Systems, Open Systems Interconnection Reference Model, Part 2: Security Architecture, ISO DIS 7498-2, Geneva, Switzerland, 1988.

[4] M. E. Hellman, "An Overview of Public Key Cryptography", *IEEE Communications Magazine,* Vol. 16, No. 6, pp. 24-32, Nov. 1978.

[5] G. J. Simmons, "Symmetric and Asymmetric Encryption", *ACM Computing Survey,* Vol. 11, No. 4, pp. 305-330, 1979.

[6] R. M. Needham and M. D. Schroeder, "Using Encryption for Authentication in Large Networks of Computers", *Communications of the ACM,* Vol. 21, No. 12, pp. 993-999, Dec. 1978.

[7] CCITT Recommendation X.509 (1988), The Directory – Authentication Framework.

[8] C. S. Neuman and J. Schiller, "Kerberos: An Authentication Service for Open Network Systems", *Proceedings Winter USENIX Conference,* Dallas, Feb. 1988.

[9] G. J. Simmons, "An Introduction to Shared Secret and/or Shared Control Schemes and Their Application", in: Simmons G. J., ed., *Contemporary Cryptology,* The Science of Information Integrity (IEEE Press, New York, 1992).

[10] C. J. Mitchell and A.Thomas, "Standardising Authentication Protocols Based on Public Key Techniques", *J. Comput. Security,* Vol. 2, No. 1, 1993.

[11] D. Mill, Internet Time Synchronization: The Network Time Protocol, RFC 1129, Oct. 1992.

[12] Rhee Man Young, *Cryptography and Secure Communications.* McGraw-Hill, 1994.

[13] Y. Desmedt, "Society and Group Oriented Cryptography: A New Concept," in *Advances in Cryptology: Proc. of Crypto'87 Lecture Notes in Computer Science*. New York: Springer-Verlag. pp. 120-27, 1987.

[14] Y. Frankel, "A Practical Protocol for Large Group Oriented Networks," in *Advances in Cryptology: proc. of Crypto'89 Lecture Notes in Computer Science*. New York: Springer-Verlag. pp. 56-61, 1989.

[15] Y. Desmedt and Frankel Y., "Threshold Cryptosystems", *in Advances in Cryptology: Proc. of Crypto'89 Lecture Note in Computer Science.* New York: Springer-Verlag. pp. 307-315, 1989.

BIOGRAPHY

L. A. Gumel is currently an M.Sc. student in the Department of Mathematics, UPM. He completed a Bachelors degree with Honours in Mathematics from Ahmadu Bello University (ABU), Zaria, Nigeria. His research areas include network optimization, cryptography and computer communication networks design.

Soo Kar Leow is attached to the Department of Mathematics, UPM. He received his B.Sc. (Hons) degree in Mathematics from the University of Malaya, M.Sc. degree in Operational Research from the London School of Economics and Political Science, England. Ph.D degree (Operations Research) from North Carolina State University, Raleigh, USA. His areas of specialization include application of operations research techniques to optimize the reliability of computer networks systems. He is currently a member of Editorial Advisory Board of the Asia-Pacific Journal of Operational Research.

A. K. Ramani received his Ph.D from Devi Ahilja University, Indore, India in 1990. Currently, he is an Associate Professor at Universiti Putra Malaysia. His areas of interest are high performance computing, networks, distributed systems, fault-tolerant computing and operating systems.