CHAPTER 4

PARAMETRIC STUDY ON AUTHENTICATION MECHANISMS FOR WEIGHTAGES

4.1 INTRODUCTION

Chapter 2 has elaborated on the versatility of security mechanisms for network. This chapter attempts to study a selected number of security mechanisms with specific reference to certain parameters that are relevant to SNS. Through the literature surveyed in chapter 2, the present research has delimited the following three parameters of network security.

Three relevant parameters have been considered for security. They are given below

1. Usability
2. Level of Security
3. Robustness

The authentication provision for security is generally not focused on the user in areas of application such as banking and government strategic e-governance. In SNS the authentication measures depend almost virtually on the users. Therefore the present research attempts to provide empirical weightages on usability with respect to security, pertaining to the above three parameters.
The first parameter relies on the computational time of SNS, particularly which is consumed by the server of SNS and the user nodes. It should be noted that the time taken by the data packets for travelling in network is not considered for the research. Hence, the time of computation for password verification either in a server or nodes alone is considered. The second parameter is dealt by several researchers globally. The level of security varies depending upon the domains of application. For instance, security application for banking, e-governance and e-commerce may demand higher security. Novice users in SNS may demand high level of usability and not much security. However, this parameter is important for the present research, as the study focuses on three different types of components namely novice users of SNS, restricted forums within SNS and secret groups within SNS. The third parameter, namely robustness is directly linked with system reliability and from a commercial view point, this is the most needed component by SNS owners.

Based on the above justification, the above mentioned three parameters are considered for the study on their strengths for security mechanism. The following five security mechanisms are delimited for the research.

1. Password based mechanism
2. Token based mechanism
3. Biometric based mechanism
4. Signature based mechanism and
5. Group key based mechanism

The present chapter elaborates on these five mechanisms that are delimited to the application of SNS. The corresponding methodologies developed by the researcher and the analytical study based on the three parameters are presented in sections 4.2 to 4.5 of this chapter.
4.2 PASSWORD BASED AUTHENTICATION MECHANISM

The most wanted and less complicated security mechanism is the simple password based mechanism. Dirk Van and Jacques (2010) have stated that it is a must to build cryptographic technique and set up a trust computing milieu to protect the information in its storage, in its transition as well as in processing. Thus it becomes essential to create an effective cyber security for the electronically stored, transmitted and processed personal information. "A major weakness in the internet is the lack of tenure of well designed and highly usable framework for reliable and privacy preserving authentication" according to the above authors.

Normally, inquiring username along with password is the traditional way of verifying the identity of a user. The same principle is used in online transaction also. In spite of its simplicity, this authentication mechanism has demerits too, particularly with respect to SNS applications.

4.2.1 Problems with Password Based Authentication

The security of this authentication system purely lies on the hardness of the password. However, if the password is hard, it would then be difficult for the user to remember it. The user may find it difficult to remember a string that consists of randomly generated characters. However, when a user selects an easily memorable name (say, her favorite things, places, persons) it becomes vulnerable to dictionary attack or brute force attack.

4.2.1.1 Parameter 1 on simple password based authentication

As per Miloslav et al (2011), the factors which have a direct impact on the security of the password is listed below,
a. Length of the password
b. Arbitrariness of characters
c. Life span of the password

If the length of the password is less, it would take fewer efforts to break it using brute force attack. If the password consists of only alphabets, it is considered as a weak password. Each character may have one of the values in the set \{a ........z\}. It is easy to guess this kind of password by simply substituting each character of the password with the alphabets a to z. If the password is chosen in such a way that it would be easy to remember, it would then not be difficult to verify and would not be lengthier. The time taken to break the password would be proportional to its arbitrariness. But, still it would be possible to safeguard the password against brute force and dictionary attacks.

In order to protect it from the attack, the user has to change the password periodically. But the lifetime needs to be decided based on the time taken for the trial attempts made by an intruder to break it. That is, the efforts taken by the intruder would be much larger than the life time of the password. In SNS, novice users wouldn’t worry about the seriousness of passing or granting their credentials to others. Therefore this issue becomes debatable.

Joanne (2011) has suggested that the method used for setting up a user name and password in social networks should include the following capabilities.

1. Bare minimum number of characters in the password.
2. Availability of reset options.
3. Option for periodical Updation.
4. CAPTCHA methods.

5. e-mail verifications.

The entire above have a direct link with SNS application domain. If the password is found to be tough, the computation for verifying the password is proportionally longer. Else, it would be vice versa, thereby making it more vulnerable to attacks.

The importance of each factor is discussed, as suggested by Joanne (2011). The length of the character should be finalized in such a way that the time and effort taken to break it should be higher than its life time. Thus it requires providing an option for periodic updating of passwords. While resetting the password again and again, it would become mandatory to verify the legitimacy of the user, by prompting the user with additional security questions. If the answers match with the already stored content, the user would then be allowed to reset her password.

To ensure that the user id is created by a human and not by any software process, Captcha’ method would be highly preferable. Email verification is also proposed to validate the user’s identity in a global scenario. Thus the basic, simple and traditional password based authentication has both advantages and disadvantages particularly for SNS application.

4.2.1.2 Parameter 2 on SAKA algorithm

Adapting a password based authentication through key exchange mechanism has its own merits and demerits. The literature on key agreement or key exchanging algorithm using password is studied and reported.

Seo and Sweeney (1999) have proposed a simple key generation algorithm based on the password selected by a user. The public values created
as in Diffie-Hellman (DH) algorithm would be shared between the users in an SNS. A typical authentication Process involved in this algorithm is demonstrated below. The communicative entities compute two unique integers namely $Q \mod q-1$ and $Q^{-1} \mod q-1$ from password $P$. These are relatively prime with $q-1$.

From the values of $Q$ and $Q^{-1}$, the session key is calculated as narrated below

Legend :

- $q$ : Prime Number
- $g$ : Primitive root of a prime number $q$
- $a$ : Random large integer
- $b$ : Random large integer
- $X, X_1$ : Intermediate value Calculated by User$_1$
- $Y, Y_1$ : Intermediate value Calculated by User$_2$

Step 1: User$_1$ choose a random large integer $a$ and calculate $X_1$ by using Equation (4.1).

$$X_1 = g^{aQ \mod q} \quad (4.1)$$

Step 2: User$_2$ choose a random large integer $b$ and calculate $Y_1$ by using Equation 4.2.

$$Y_1 = g^{bQ \mod q} \quad (4.2)$$

Step 3: Mutually Exchange the Values of $X_1, Y_1$
Step 4: User\(_1\) calculated \(Y\) by using Equation (4.3).

\[
Y = Y_1^{Q-1} \pmod{q}
\]  

(4.3)

Step 5: User\(_2\) calculates \(X\) by using Equation (4.4).

\[
X = X_1^{Q-1} \pmod{q}
\]  

(4.4)

Step 6: User\(_1\) calculate \(Key_1\) by using Equation (4.5).

\[
Key_1 = Y^a \pmod{q}
\]  

(4.5)

Step 7: User\(_2\) calculates \(Key_2\) by using Equation (4.6).

\[
Key_2 = X^b \pmod{q}
\]  

(4.6)

Step 8: User\(_1\) computes the intermediate value using Equation (4.7).

\[
(Key_1)^Q \pmod{q}
\]  

(4.7)

Step 9: User\(_2\) computes the intermediate value using Equation (4.8).

\[
(Key_2)^Q \pmod{q}
\]  

(4.8)

Step 10: Mutually Exchange the values calculated in the step 8 and step 9.

At both end user verifies the key by take the inverse of the received value and compare it against its own key. If the values are found equal then the mutual authentication is completed.
Subsequent to this development, Iuon-Chang et al (2000) have proposed a Security Enhancement for the Simple Authenticated Key Agreement Algorithm (SAKA). The authors have identified three weaknesses in the earlier protocol proposed by Seo and Sweeney (1999). SAKA didn’t provide any option to confirm the identity of the user. It is still vulnerable to dictionary attack. It didn’t provide Perfect Forward Secrecy (PFS). To overcome the above lacuna, Iuon-Chang et al (2000) have proposed an enhancement to SAKA.

4.2.1.3 Parameter 3 on security enhancement for the simple authenticated key agreement algorithm

In this enhanced protocol, the keys are computed as stated in SAKA as narrated below,

Key₁ at User₁’s end

Key₂ at User₂’s end

In the formal algorithm, the intermediate values which are calculated using Key₁ and Key₂ are exchanged. In the enhanced model, the author has exchanged the public values $K₁$, $K₂$ calculated through Equations (4.8) and (4.9) as explained below.

User₁ computes \[ K₁ = Y^{θ-1} \mod q \] (4.9)

Similarly User₂ computes \[ K₂ = X^{θ-1} \mod q \] (4.10)

The validation the key is carried over as shown below.

User₁ verify the $K₂$ by using Equation (4.11).
Similarly User\(_2\) verify the value of \(K_1\) by using Equation (4.12).

\[
K_1 = (\frac{b^{\theta-1}}{g}) \mod q 
\]  

(4.12)

On concurrence, it would be concluded that session Key\(_1\) and Key\(_2\) are equal and authentication is completed.

4.2.1.4 Parameter 4 on cryptanalysis on the security enhancement for the modified SAKA algorithm and its remedial measures

Kou-Min et al (2010) have done cryptanalysis on this security enhancement of the modified SAKA protocol. It was reported that for the verification purpose, two more steps are introduced. The private random number namely \(a\) and \(b\) selected at each end was included to calculate \(Y_2\) and \(X_2\) as demonstrated below.

User\(_1\) calculates \(Y_2\) from \(Y\) by using Equation (4.13).

\[
y_2 = (aY)^{\theta-1} \mod q
\]

(4.13)

User\(_2\) calculates \(X_2\) from \(X\) by using Equation (4.14).

\[
x_2 = (bX)^{\theta-1} \mod q
\]

(4.14)

Mutually exchange the values of \(x_2, y_2\)

User\(_1\) tried to derive the value of \(b\) by using Equation (4.15) and (4.16).
\[ b = \frac{(x_2)^Q}{X} \mod q \]  
\[ X_2 = (bx)^{Q-1} \]  

By substituting (4.16) in (4.15).

It becomes Equation (4.17).

\[ b = \left( \left( \frac{b}{x} \right)^{Q-1} \right)^Q / X \]  
\[ = \left( \frac{b}{x} \right)^{\frac{Q}{Q-1}} / X = b \]  

Similarly by receiving \( Y_2 \), User_2 tried to derive the value of ‘a’ by using Equation (4.19).

\[ a = \left( y_2 \right)^\theta / Y \mod q \]  

The Right hand side of Equation (4.20) is calculated from the value ‘b’ which is derived by using the Equation (4.18). It is compared with its Left hand side. If it found equal then the authentication is completed.

\[ Y \mod q = g^b \]  

Similarly it is verified at the other end using the Equation (4.21).

\[ g^a = X \mod q \]  

Since the User_1 knows the secret key of the other end and vice versa,
Session key is verified by using the Equation (4.22).

\[ K_{AB} = g^{ab} \]  

(4.22)

KoU – Ming et al (2010) have identified the following security flaws in this method,

1. Backward replay attack is possible and

2. Offline password guessing attack is also possible.

In the above mathematical demonstrations the merits and demerits of password based security mechanisms with reference to SNS have been discussed eventually. Based on the merits and demerits discussed on the above parameters, weightage values have been empirically arrived and presented in Table 4.1.

Table 4.1 Empirical weightage values for various levels of password based authentication

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Authentication Mechanism</th>
<th>Level</th>
<th>Weightages on Usability</th>
<th>Security</th>
<th>Robustness</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Password chosen such a way to produce high security</td>
<td>Level1</td>
<td>0.72</td>
<td>0.1</td>
<td>0.1</td>
<td>0.92</td>
</tr>
<tr>
<td>2</td>
<td>SAKA</td>
<td>Level2*</td>
<td>0.6</td>
<td>0.2</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>3</td>
<td>Enhanced SAKA</td>
<td>Level3**</td>
<td>0.66</td>
<td>0.3</td>
<td>0.1</td>
<td>1.06</td>
</tr>
<tr>
<td>4</td>
<td>Cryptanalysis on Enhanced SAKA</td>
<td>Level4***</td>
<td>0.5</td>
<td>0.8</td>
<td>0.2</td>
<td>1.50</td>
</tr>
</tbody>
</table>

* Presuming 20% of more computation than level1
** Presuming 10% of more computation than level1
*** Assuming 30% of more computation than level1
The volume of computation required to verify the password and to calculate the keys based on it have downside impact on its usability. But it has positive impact on security. On the other hand, robustness depends on the freshness of the values submitted for key calculation. From the social survey conducted and analyzed (presented in Chapter 3) and based on the performance of these authentication mechanism, these weightages have been arrived. The most preferred mechanism (with highest weightage) is highlighted in the respective table given above.

Based on the above findings, the researcher has attempted to propose a reliable authentication model for SNS. This model has been designed based on the above levels and has been implemented, executed and validated. They are elaborated in the next chapter.

4.3 TOKEN BASED AUTHENTICATION

Catherine et al (2009) have elaborated three different types of two factor authentication models. Lindell (2007) has stated that one time password or access code is generated in three different ways. Following are the operations that are executed after submission of user name and password.

1. User has to press the button on the device, OR
2. User has to insert a smart card to a card reader and then press the button, OR
3. User has to insert a smart card followed by entering a pin number and then press the button.

The user would have to provide a One Time Password (OTP) or secret code to proceed further. Banking application is an example for this kind of authentication.
To practice this type of authentication system as mentioned in option1, a device becomes mandatory to generate OTP. For the next option, the device must be attached with a card reader to validate the user. For the third option, the system must have a device to accept the card as well as a PIN to validate and to generate an OTP. This shows that this scheme requires additional storage. If it is applied in any e-banking environment, it would then become mandatory to design a specific framework with hardware for user interface.

The success of the authentication method purely depends on the degree of ease it offers to the users. Such devices should not be a hindrance to the users of SNS. Catherine et al (2009) have indicated different levels of users in their survey. According to the authors, most of the users need to know every step of the procedures. Based on the devices, empirical weightages have been arrived at and presented in Table 4.2.

**Table 4.2  Empirical weightage values for various levels of token based authentication**

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Authentication Mechanism</th>
<th>Level</th>
<th>Weight ages on</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Usability</td>
<td>Security</td>
</tr>
<tr>
<td>1</td>
<td>Press the button on the device to get onetime Password</td>
<td>Level1</td>
<td>0.173</td>
<td>0.4</td>
</tr>
<tr>
<td>2</td>
<td>Insert a smart card &amp; press the button</td>
<td>Level 2*</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>3</td>
<td>Insert a smart card, enter the PIN and press the button</td>
<td>Level 3**</td>
<td>0.05</td>
<td>0.6</td>
</tr>
</tbody>
</table>

* Presuming approximately 50% reduction in usability due to introduction of smart card

** Presuming approximately 50% reduction in usability due to introduction of smart card and PIN
The computation required for generating one time password and its device dependent nature drastically affects its usability. However, it has positive impact on security proportionally. Robustness depends on the freshness of the values submitted for password calculation. The weightages have been arrived from the social survey conducted, analyzed and based on the performance of authentication mechanism.

4.4 BIOMETRIC BASED AUTHENTICATION

When the mutual authentication depends on bio-metric features to validate user’s identity, huge storage space is required to hold the biometric templates of them. This provides high level of access control. As per the findings of Venkata SubbaReddy et al (2009) user would be in need of a special device to submit the bio-metric response like web camera to present his facial response, or microphone for voice response, or a scanner for iris response or finger print reader. Users and devices are pervasive in nature. If a system fails to provide such facilities, it would then not be possible to carry over the authentication process. If an authentication process considers device addresses or tokens of devices as a metric to validate the user, user’s accessibility would then be restricted. Besides, it forces the users to hold the devices to access the system. Smart card/RFID tags are two common examples. In case of third party interview, if the third party, such as an e-customer support employee is compromised, the system becomes vulnerable to impersonation attack. This additional complicated procedure will make a system more unusable. Based on the levels of difficulty, weightages have been arrived empirically. The values are presented in Table 4.3.
Table 4.3  Empirical weightage values for various levels of biometric based authentication

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Authentication Mechanism</th>
<th>Level*</th>
<th>Weight ages on</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Usability</td>
<td>Security</td>
</tr>
<tr>
<td>1</td>
<td>Facial Response</td>
<td>Level1</td>
<td>0.05</td>
<td>0.4</td>
</tr>
<tr>
<td>2</td>
<td>Finger Print Impression</td>
<td>Level2</td>
<td>0.05</td>
<td>0.5</td>
</tr>
<tr>
<td>3</td>
<td>Irish Response</td>
<td>Level 3</td>
<td>0.05</td>
<td>0.4</td>
</tr>
<tr>
<td>4</td>
<td>Voice Response</td>
<td>Level 4</td>
<td>0.05</td>
<td>0.4</td>
</tr>
<tr>
<td>5</td>
<td>Palm Response</td>
<td>Level 5</td>
<td>0.05</td>
<td>0.5</td>
</tr>
</tbody>
</table>

* Presuming the intensities of cumbersome involved through physical efforts, starting empirical value for usability is 0.05 that is comparable with previous tables.

The computation required for verifying the identity of a user and its device dependent nature is a drawback on its usability. The false positive rate also increased for situations like face variation happens on the mood of a person. Voice changes due to problems in throat. They have proportionally a positive impact on the security. The robustness depends directly on the original values submitted for key calculations. The weightages have been arrived, based on the analysis of the social survey conducted and on the performance of the authentication mechanisms.

4.5 SIGNATURE BASED AUTHENTICATION

Digital signature is a proven and highly secured authentication method that provides the following cryptographic services.

1. Message integrity
2. Data origin authentication
3. Non repudiation
4.5.1 Parameter 1 on Message Integrity

The signature is normally created by encrypting the message digest using private key of the sender. The Digest is generated by using hash functions. It will be verified at the receiving end by calculating the hash value of the message received. If both the values are equal it would be assumed that the message has reached the receiver without any alteration during in its transition. This is vividly presented in Figure 4.1.

4.5.2 Parameter 2 on Proof for Data Origin

In this process, the data origin is verified by encrypting the message with sender’s private key, as private key is known only to the owner. If the message is encrypted by a private key, indirectly means that the origin of the message is the owner of that key. This process is clearly depicted in Figure 4.2.
4.5.3 Parameter 3 on Proof for Non Repudiation

A sender may send a message with his/her signature. Afterward he/she might deny it. To overcome this, the sender is instructed to send the message through a notary. This process is clearly depicted in Figure 4.3.

![Figure 4.3 Digital signature as a proof for non repudiation](image)

4.5.4 Parameter 4 on Diffie-Hellman Key Agreement Algorithm

Two users of an SNS can create a symmetric session key without the help of a Key Distribution Centre (KDC). Before establishing a symmetric key, the two users need to choose two numbers namely q and g.

- q is a large prime number, \( g < q \) and g is a primitive root of q. These two elements are announced as public. The process is explained below.

Step1: User1 choose a random number \( x \) such that \( 0 \leq x \leq q - 1 \) and calculate its public key by using Equation (4.23)

\[
X_1 = g^x \mod q \tag{4.23}
\]
Step 2: User 2 choose a random number $y$ such that $0 < y < q$ and calculate his public key by using Equation (4.24).

$$Y_1 = g^y \mod q$$ (4.24)

Step 3: Users exchange the public keys mutually.

Step 3: User$_1$ calculates the session key by using Equation (4.25).

$$K_1 = (Y_1)^x \mod q$$ (4.25)

Step 4: User$_2$ calculates the session key by using Equation (4.26).

$$K = (X_1)^y \mod q$$ (4.26)

Security Analysis of Diffie Hellman Protocol has been demonstrated by Jean-Francois and Anton (2000) and proved that, it is possible to have the following two types of attacks, namely,

I. Discrete Logarithm Attack and

II. Man in the Middle Attack

The two types of attacks are explained below:

I. Discrete Logarithm Attack

The process is explained through a mathematical model. $X_1$ and $Y_1$ are exchanged in an open network. If an intruder is able to find $x$ and $y$ using $X_1$ and $Y_1$, it becomes so simple for the intruder to calculate the session key.
To make the protocol withstand this kind of attack the Forouzan B.A. (2007) has proposed the following solution,

1. Choose $q$ as a large prime number
2. Select $q$ in such a way that $(q-1)$ has at least one large prime factor
3. The value of $x$ and $y$ must be used only once accordingly, weightages can be arrived at.

II. Man in the Middle Attack

An intruder can impersonate both the sender and receiver of the SNS, by creating two keys, one in each direction. The process is demonstrated below using Figure 4.4.

**Figure 4.4 Man in the middle attack in DH protocol**
The main drawback of DH protocol is its inability to handle Man in the Middle Attack. Since the identities of the sender and receiver are not included in the message, they are not verified. Thus, it would fail to authenticate the users while exchanging the keys. It is demonstrated in Figure 4.4.

4.5.5 Parameter 5 on Station to Station Key Agreement Protocol

It is a method which is based on DH Protocol. It uses digital signatures with public key certificate as a proof for data origin. The process is explained in Figure 4.5.

Figure 4.5 Station to station key agreement protocol
The various steps involved in the process are explained below.

Step 1 : User₁ after choosing his private key calculates his public key

Step 2 : Sends his public key to User₂

Step 3 : User₂ calculates his public key

Step 4 : User₂ calculates the session key using User₁’s public key

Step 5 : User₁ sends a signed message (Name of the receiver, public Key received, public key of his own)

The public key can then be certified by a trusted third party. In the case of SNS it can be done by the SNS’s owner, called the Central Agency (CA).

Step 6 : By receiving User₂’s public key, User₁ can calculate the session key

Step 7 : Verify User₂’s signature

Step 8 : Send signature generated on the message (Name or ID of receiver, public key received, public key of his own). This public key could also be certified by CA. (see Step 5).

Step 9 : User₂ verifies the User’s identity by verifying its signature

Step 10 : Process is completed

Security Analysis of Station to Station Protocol has made and identified that the protocol prevents man in the middle attack. The intruder cannot pretend to be an authorized sender of SNS, as the sender’s private key,
is not known to him. The public key has also been passed in the form of certificates issued by trusted authorities (say SNS Owner, or CA).

Normally, one way functions are used in digital signature schemes. As per the findings of Harn (1997) digital signatures generated without using one way function, can easily be forged. He has proposed a signature scheme. Here four session keys are calculated in one attempt. The signature is verified with two modular equations. He has concluded that this signature scheme would be suitable for signing the DH public keys. Based on this argument, weightages have been arrived.

4.5.6 Parameter 6 on Authenticated Key Exchange Protocol

Harn and Lin (2001) have proposed a key exchange protocol that provides authentication without using one way hash function. This method generates four session/shared keys between the users of SNS by using two short term private keys and two short term public keys. Only three keys are used, in order to provide Perfect Forward Secrecy. The total computation time required is very high. This enhanced version of authenticated key exchange protocol suggested by Harn and Lin (2001) has been further enhanced by Hwang et al (2003). However appropriate weightages have been arrived at for the above protocol also.

\[
q \quad : \quad \text{Large Prime Number } g \rightarrow \text{Primitive root of } q
\]

Step 1.1 : User\(_1\) choose a random number \(x\) such that \(0 \leq x < q\) and calculate its long term public key by using Equation (4.23)

Step 1.2 : User\(_2\) choose a random number \(y\) such that \(0 < y < q\) and calculate its long term public key by using Equation (4.24)
Step 2.1 : User_1 choose \( R_{a1}, R_{a2} \) as a short term secret keys and calculate his long term public Keys as \( U_{a1}, U_{a2} \) by using the Equation (4.27),(4.28) such that \( U_{a1}, U_{a2} \)

\[
U_{a1} = g^{R_{a1}} \mod q
\] (4.27)

\[
U_{a2} = g^{R_{a2}} \mod q
\] (4.28)

Step 2.2 : User_2 choose \( R_{b1}, R_{b2} \) as a short term secret keys and calculate his long term public Keys as \( U_{b1}, U_{b2} \) by using the Equation (4.29),(4.30) such that \( U_{b1}, U_{b2} \)

\[
U_{b1} = g^{R_{b1}} \mod q
\] (4.29)

\[
U_{b2} = g^{R_{b2}} \mod q
\] (4.30)

Step 3.1: User_1 calculate \( W_a \) by using the Equation (4.31).

\[
W_a = x - (U_{a1}R_{a1} + U_{a2}R_{a2}) \mod q - 1
\] (4.31)

Step 3.2: User_2 calculate \( W_b \) by using the Equation (4.32).

\[
W_b = y - (U_{b1}R_{b1} + U_{b2}R_{b2}) \mod q - 1
\] (4.32)

Step 4.1: User_1 send the message as in Equation (4.33) to User_2.

\[
U_{a1}, U_{a2}, W_a, \text{Cert}(Y_a)
\] (4.33)

Step4.2: User_2 send the message as in Equation (4.34) to User_1.

\[
U_{b1}, U_{b2}, W_b, \text{Cert}(Y_b)
\] (4.34)

Step 5.1: User_1 verify the signature by Equation (4.35).
\[ g^y = g^{R_{bl} U_{bl} R_{b2} U_{b2} w_b \mod q} \]  \hspace{1cm} (4.35)

\[ g^{R_{bl}} = U_{bl} \]  \hspace{1cm} (4.36)

\[ g^{R_{b2}} = U_{b2} \]  \hspace{1cm} (4.37)

\[ Y = g^y \mod q \]  \hspace{1cm} (4.38)

By substituting Equation (4.36),(4.37),(4.38) in Equation (4.35), it becomes Equation (4.39).

\[ Y_1 = U_{bl}^{U_{bl}} U_{b2}^{U_{b2}} g^{w_b \mod q} \]  \hspace{1cm} (4.39)

Step 5.2: User₂ verify the signature by Equation (4.40)

\[ g^x = g^{R_{al} U_{al} R_{a2} U_{a2} w_a \mod q} \]  \hspace{1cm} (4.40)

\[ g^{R_{al}} = U_{al} \]  \hspace{1cm} (4.41)

\[ g^{R_{a2}} = U_{a2} \]  \hspace{1cm} (4.42)

\[ g^x = X \]  \hspace{1cm} (4.43)

By substituting Equation (4.41),(4.42),(4.43) in Equation (4.40), it becomes Equation (4.44).

\[ X_1 = U_{al}^{U_{al}} U_{a2}^{U_{a2}} g^{w_a \mod q} \]  \hspace{1cm} (4.44)

Step 6.1: User₁ compute the session keys as shown Equations (4.45),(4.46),(4.47), (4.48).
4.5.7 Parameter 7 on the Security of an Enhanced Authenticated Key Exchange Protocol

Through their enhancement, the authors have checked for reduction in the computational cost to improve efficiency by using an ex-or operation. The modification proposed in steps 3.1, 3.2, 5.1 and 5.2 of earlier protocol are explained below.

Step 3.1: User 1 Calculate $W_a$ by the following Equation (4.49).

$$W_a = x (U_{a1} \oplus U_{a2}) + R_{a1} \mod (q-1) \quad (4.49)$$

Step 3.2: User 2 Calculate $W_b$ by the following Equation (4.50).

$$W_b = y (U_{b1} \oplus U_{b2}) + R_{b1} \mod (q-1) \quad (4.50)$$

Step 4.1: User 1 send $U_{a1}, U_{a2}, W_a, Cert(Y_a)$ to User 2.

Step 4.2: User 2 send $U_{b1}, U_{b2}, W_b, Cert(Y_b)$ to User 1.

Step 5.1: User 1 verify the signature by Equation (4.51).
Since \( g^y = Y_1 \)

Substituting the Equation (4.52) in the Equation (4.51), resulted to Equation (4.53).

\[
g^{w_b} = g^{y_{(U_b1 \oplus U_b2)} R_b \mod q} \tag{4.53}
\]

Step 5.2: User 2 verify the signature by Equation (4.54).

\[
g^{w_a} = g^{x_{(U_a1 \oplus U_a2)} R_a} \tag{4.54}
\]

\[
g^x = X_1 \tag{4.55}
\]

Substituting the Equation (4.55) in the Equation (4.54) becomes Equation (4.56).

\[
g^{w_b} = g^{y_{(U_b1 \oplus U_b2)} U_{b1} \mod q} \tag{4.56}
\]

Four keys are calculated as explained above, out of which only three keys are used as session keys. Hwang et al (2003) has advised to omit \( K_1 \). It was demonstrated that how an adversary could trace the session key \( K_{ab} \) with the knowledge of \( K_1 \). This was considered as a root cause that affects the Perfect Forward Secrecy of the protocol. The authors have defended the computational efficiency of their protocol with that of the earlier protocol. There were some security flaws that were identified by Min-shiang et al (2004). They have reported that forgery attack could be mounted in the enhanced authentication key exchange protocol that was proposed by Hwang et al (2003). Based on the above issues, weightages have been arrived.
4.5.8 Parameter 8 on an Enhanced Authenticated Key Exchange Protocol

The process of forgery attack on enhanced protocol is explained in Figure 4.6.

Step 1: The message contains long term public key Certificate, signature and short term Public keys of legitimate sender send to the receiver.

Step 2: The message is stolen and modified with intruder’ Parameters and append the certificate of the legitimate sender and transmitted to the receiver.

Step 3: The message contains the details of the receiver as specified in Step 1.

Step 4: It is also stolen by the intruder, modified and transmitted to the sender.

Figure 4.6 Forgery attack in enhanced authenticated key exchange protocol
An intruder can impersonate both the users of SNS. The forgery attack could be detected after computing the keys by this method. In case of attacks, the system would have unnecessarily wasted its computational time. Due to this forgery attack, the above said protocol is also vulnerable to Denial of Service (DoS) attack.

Min-shiang et al (2004) have proposed a remedial measure for this forgery attack by enhancing the procedure to calculate signature as shown in Equation (4.57)

\[ W_a = x(U_{a1} \oplus U_{a2}) + R_{a1} \ast U_{a2} \mod(q-1) \]

(4.57)

While User 2 receives the message \( \{U_{a1} \cdot U_{a2} \cdot W_a \cdot y_a\} \)

She verifies the message by using Equation (4.58) as given below.

\[ w_a = x(U_{a1} \oplus u_{a2}) \cdot R_{a1} \cdot U_{a2} \mod_q \]

(4.58)

Since \( x_{a1} = g \mod_q \)

(4.59)

\[ U_{a1} = g \cdot R_{a1} \mod_q \]

(4.60)

The Equation (4.61) is derived by substituting Equation (4.59), (4.60) in the Equation (4.58).

\[ x_{a1}^{(U_{a1} \oplus U_{a2})} \cdot U_{a2} \mod_q \]

(4.61)

It could not to prevent the forgery attack. Chou-Chan et al (2005) have proved that Hwang et al (2003) protocol is insecure, by having done the cryptanalysis on it. An attacker could forge the sender as well as the receiver, if he gets old messages from legitimate users. (The old users of SNS).
4.5.9 Parameter 9 on an Enhanced and Secure Protocol for Authenticated Key Exchange

By using the result of earlier cryptanalysis, Siva Prasad et al (2008) have proposed a way to overcome the forgery attack. They have proposed an interactive way of authenticating message exchanges as discussed below and shown in Figure 4.7. The main drawback of this proposal is that, the signature is once again calculated by using hash function. This is explained in the next parametric study. The weightages are suggested by this current method also.

user A

\[ M_a = \{ U_{a1}, U_{a2}, U_a, Cert(Y_a) \} \]

user B

\[ M_b = \{ U_b, U_{b1}, U_{b2}, S_b, Cert(Y_b) \} \]

\[ S_a = \{ M_b \} \]

Figure 4.7 Interactive way of authenticating message exchanges

User\(_1\) verifies the certificate \((Y_b)\) received through the message \(M_b\).

User\(_1\) verifies the message by using Equation (4.62).

\[
g^*_{b, U_b} = (U_{b1}, U_{b2}, U_{a1}, U_{a2}, Cert(Y_a)) Y_b U_{b} \mod q \tag{4.62}
\]

If it does not match, the process will be stopped. Otherwise User\(_1\) computes \(S_a\) by using the Equation (4.63) as given below,
\[ S_a = R_a^h \left( U_a U_{a1} U_a2 U_{b1} U_{b2} \text{Cert}(y_a) + X_a U_a \mod q \right) \]  \hspace{1cm} (4.63)

User₁ then computes the session keys and sends the response to User₂.

While receiving the message \( M_a \) User₂ Verify the signature by the Equation (4.64).

\[
\begin{align*}
S_a & = U_a^h (U_a U_{a1} U_a2 U_{b1} U_{b2} \text{Cert}(Y_a)) \mod q \\
\end{align*}
\]  \hspace{1cm} (4.64)

If the validation fails, the key agreement Protocol stops its process. Else, it computes the shared session keys.

It is also proved that both the users check the messages \( M_b \) and \( M_a \) which are generated based on signatures \( S_b \) and \( S_a \). Thus the proposed Protocol is found to be secured from the forgery attack. But in the Protocols proposed, enhanced and analysed Harn (1997), Harn and Lin (2001) Hwang et al (2003), Min-Shiang et al (2004) and Chou-Chan et al (2005), it is seen that the signatures were generated without using one way hash function. But Siva Prasad et al (2008) have proposed a remedial measure for the security flaw identified in the above mentioned protocol with hash functions.

Except in the remedial measure discussed by Siva Prasad et al (2008) in all other cases, the keys are calculated before verifying the authenticity of the entities. Weightages based on the above arguments have been arrived at. All the weightages of the 9 parameters are displayed in Table 4.4.
Table 4.4 Empirical weightage values for various levels of signature based authentication

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Authentication Mechanism</th>
<th>Level</th>
<th>Weight ages on Usability</th>
<th>Security</th>
<th>Robustness</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Signature using Hash Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Proof for message integrity</td>
<td>Level 1*</td>
<td>0.36</td>
<td>0.4</td>
<td>0.2</td>
<td>0.96</td>
</tr>
<tr>
<td>2</td>
<td>Data origin authentication</td>
<td>Level 2*</td>
<td>0.36</td>
<td>0.5</td>
<td>0.2</td>
<td>1.06</td>
</tr>
<tr>
<td>3</td>
<td>Proof for non repudiation</td>
<td>Level 3**</td>
<td>0.18</td>
<td>0.5</td>
<td>0.2</td>
<td>0.88</td>
</tr>
<tr>
<td>4</td>
<td>Diffe Hellman key exchange</td>
<td>Level 4***</td>
<td>0.36</td>
<td>0.3</td>
<td>0.1</td>
<td>0.76</td>
</tr>
<tr>
<td>5</td>
<td>Station to station protocol</td>
<td>Level 5 ****</td>
<td>0.18</td>
<td>0.4</td>
<td>0.1</td>
<td>0.68</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Signature without using Hash Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Authenticated key agreement</td>
<td>Level 6+</td>
<td>0.54</td>
<td>0.3</td>
<td>0.1</td>
<td>0.94</td>
</tr>
<tr>
<td>2</td>
<td>The Security of an enhanced authenticated key exchange protocol</td>
<td>Level 7++</td>
<td>0.63</td>
<td>0.4</td>
<td>0.1</td>
<td>1.13</td>
</tr>
<tr>
<td>3</td>
<td>On enhanced authenticated key exchange protocol</td>
<td>Level 8++</td>
<td>0.63</td>
<td>0.5</td>
<td>0.1</td>
<td>1.23</td>
</tr>
<tr>
<td>4</td>
<td>An enhanced and secure protocol for authenticated key exchange</td>
<td>Level 9++</td>
<td>0.63</td>
<td>1.0</td>
<td>0.2</td>
<td>1.83</td>
</tr>
</tbody>
</table>

* Presuming half of usability compared with simple password (Table 4.1 of Password based Authentication Mechanism)
** usability further reduced in Level 2
*** Usability similar to Level 1
**** Usability similar to Level 3
+ Increase usability by 50% of Level1
++ Increase usability by 75% of Level1
It is inferred from the above table that the computational complexity in generating the signature at the sender’s side and verifying the same at the receiver’s side may have a downside impact on usability. However, it would have positive impact on security. The robustness component is purely based on the freshness of short term key values submitted to generate the signature. Another important authentication technique that may influence secret groups of SNS is presented in the next part.

4.6 GROUP KEY BASED AUTHENTICATION

Normally, a group represents a small number of people assembled to share common agenda. In Social Networks, set of like-minded users can be formed into groups so that members can post their messages, or chat with each other. This concept is introduced in Chapter 5 as secret groups.

Multicast communication provides an easy and simple way to establish a group communication. The information provided by the users should be accessed in a controlled manner. User may restrict the access to certain groups like class mates and office colleagues. The multicast communication is found to be deficient in providing controlled access to messages passed or exchanged in groups. It does not have any inherent security mechanism for controlling the access of messages, exchanged by non members of the group (say an intruder). To preserve the confidentiality and for maintaining controlled access, it is suggested to encrypt the message before transmitting it. The security of the encryption process purely depends on its cryptographic keys. In group communication, symmetric key crypto system is preferred. The key used for symmetric key encryption of messages that is to be exchanged is called as Group Key. The members can use the Group Key (GK) for both encryption and decryption.
This might assure the confidentiality of the messages. To control the access, rekey operation is suggested. It is always expected to hide the messages which were exchanged previously to the newly joined members. Similarly, it is expected that future messages are not revealed to the relieved members. Thus, it is clearly demonstrated that it is mandatory to execute the rekey process for every member joined as well as for those members who have left.

When a registered member of a Group likes to enter the Group’s site, the key has to be modified and informed to the old as well as new members. It is a simple process, since the new key can be encrypted using the old group key and can also be forwarded to the old member. By using only one multicast and unicast a key can be refreshed.

However, in the case of members leaving the operation, it is not possible to encrypt the new key with the old key, as the old key is already known to the relieved member. Thus, it is a crucial point to be considered before designing or proposing a novel method. Basically, Group Key Communication Mechanism is classified into three types as mentioned below.

i. Centralized Group Key Management Protocol

ii. Decentralized Group Key Management Protocol and

iii. Distributed Key Management Protocol

In the centralized group key management schemes, the group moderator will control the entire activities of the group. In a Decentralized environment a large group is divided into subgroups, so that each may have its own moderator. Each sub group can work independently. But in the Distributed Key management Protocols, there is no centralized control in the group. The members of the group themselves do the key generation. The identity of the user must be verified before allowing the member to participate in group communication. This procedure is explained in the Figure 4.8.
Rafaeli and Hutchison (2003) have done a survey on group key communication mechanisms that exist under the classification stated above. Each method has both its advantages and disadvantages. They are discussed below so as to arrive at weightages for group key based authentication.

![Figure 4.8 Procedure for validating the user in groups](image)

From the Literature survey, it is reported that Rafaeli and Hutchison (2003) have summarized the Generalized Group Key Management Protocols. Several issues for the purpose of arriving at weightages are discussed below.

Wong et al (2000) and Wallner et al (1999) have proposed the Logical Key Hierarchy (LKH) Protocol. According to this proposal, key distribution centre has to maintain a tree of keys. Each member of a node is
assured with a Key Encryption Key (KEK). The key held by the root node would be the group key. For a balanced tree structure, where \( \log_2 n \) is the height of the tree (where \( n \) is number of nodes). Each member of the tree can store to a maximum, \((\log_2 n+1)\) number of keys in it. Join operation needs to rekey the keys of all the node from the new leaf node’s parent in the path of the root. Thus, the size of a rekeying message for a balanced tree has at most two \((\log_2 n)\) keys.

If any member leaves the group, the keys in all the nodes from the left member parent to the root node has to be updated. Here, rekey message would be encrypted using respective node’s children’s key. For the leaf node which is left, no key can be used for encryption. Thus, it is suggested that the key may be encrypted with other member’s leaf node’s KEKs. Following that LKH, an additional protocol has been proposed by Wald Vogel et al (1999). It is similar to LKH, but, instead of sending the rekey to all the existing members in the group, it is suggested to send the changes through a one way function.

From the old key, members can create a new key. Following this, an one way function tree is proposed by Mc Grew and Sherman (1998). Here, the parent node’s KEKs are generated by using Equation (4.65) from its children using a one way function and mixing it together by using a mixing function ‘z’, as demonstrated below.

\[
K_i = f(z(KEK_{\text{left}}(i)), z((KEK_{\text{right}}(i))))
\]  
\[(4.65)\]

\(K_i\) = key of \(i^{\text{th}}\) node in the left

\(KEK_{\text{left}}\) : KEY Encryption Key of left child of node ‘\(i\)’

\(KEK_{\text{right}}\) : KEY Encryption Key of Right child of node ‘\(i\)’
Subsequently, Canetti et al (1999 a) have proposed a one way function chain tree. It is suggested to introduce a pseudorandom generator to generate a key instead of one way function. Canetti et al (1999 b), have also proposed another algorithm named as Hierarchical a-ary tree with clustering. In this proposal, each leaf node of a tree is assigned with a cluster of size n. These clusters are sub groups of the main group, where the cluster members can share the common Key Encryption Key (KEK). In addition to this, each member within the group is assigned with $K_i$ that can be calculated using pseudo random function $f_r(r = \text{random seed})$ as shown in the Equation (4.66).

\[ K_i = f_r(i) \quad (4.66) \]

When a member leaves, a new KEK is issued to cluster. The new KEK is encrypted with individual’s $K_i$ of its member. Wald Vogel et al (1999) have established another model named as centralized Flat Table. Here the table has one entry for the Traffic Encryption Key (TEK) and $2w$ more entries for KEKs where $w$ is the number of bits in the member id. Normally each bit is associated with one key for each possible state of the bit. When a member leaves the Group, all the keys known to the user would be modified or changed and the KDC will send the rekey message.

Perrig et al (2001) have proposed the Efficient Large group Key (ELK) protocol. Here the keys are generated and manipulated by pseudo random functions. Four different keys are generated for unique purposes. Each has its own purpose like, to encrypt the key update message, to generate hints, to update key nodes and to generate $n_1$ and $n_2$ by which $K_i$ is calculated.

In a Distributed Key management algorithm, group controller is not available to monitor the activities of the group. Group keys are generated by any one member of a group or by all the members by contributing to generate
a key. For generating a key, sufficient computational facility must be available in every node and also expected to be fault tolerant. It is mandatory that the key generator node to know about live member details. Few of the famous schemes under this distributed key generation strategy is explained one after another.

GDH (Group Diffie-Hellman) is an extension of DH Protocol of Steiner et al (1996). It is used to exchange the keys among the members of the Group. Each member will raise the intermediate value by its own secret key. Thus the last member can calculate the session key by using Equation (4.67).

\[ K = (g^{x_1 \ldots x_n}) \mod q \] (4.67)

where \( g \) is the primitive root of the Prime Number \( q \). The computational cost is directly proportional to the number of members in this case.

In Octopus Protocol (Becker and Wille (1998), a large group is divided into four sub groups. All the sub groups calculate its Group DH based on intermediary key. The sub groups can exchange their intermediate values like:

\( I_a \) → Intermediate DH Group Value of Group A
\( I_b \) → Intermediate DH Group Value of Group B
\( I_c \) → Intermediate DH Group Value of Group C
\( I_d \) → Intermediate DH Group Value of Group D

Leaders of the group A B C D are exchanged their intermediate values. Now all leaders calculate \( a^{I_a I_b I_c I_d} \). The members can calculate the group key with the value so computed.
In Conference Key Agreement Boyd(1997), Group Key is calculated with the help of MAC and hash function as given below.

\[ N \rightarrow \text{group size} \]

\[ N_i \rightarrow \text{contribution of group member } i \]

The key is then calculated by using Equation (4.68).

\[ K = f(N_1, h(N_2), h(N_n)) \] (4.68)

Here Public Key Infrastructure (PKI) is used to broadcast the one member’s contribution to all other members of the group.

In the Distributed Logical Key Hierarchy Rodeh et al (2000), the group is divided into two sub groups named as Left tree and Right tree. The sub groups agree on mutual key. The sub key’s mutual agreement continues and further subdivided towards each bottom level, until the leaf node is reached. Here, members have to store \( \log_2 n \) rounds and have to store \( \log_2 n \) keys.

Distributed One Way Function Tree (Dondeti et al 1999) is similar to the one way function tree algorithm proposed by McGraw and Sherman (1998). Every group member is considered as a trusted entity with access control and group key generation privileges. Each member generates its own key and blinds it and send it to the sibling. The member’s blinded version of the key is sent to members who are all exist in his path.

In Diffie Hellman Logical Key Hierarchy (Perrig 1999 and Kim et al 2000) protocol each member calculates its key from its children’s key value, as shown in the Equation (4.69).

\[ K_i = \alpha^{k_i} \alpha^{k_r} \mod q \] (4.69)
Distributed flat table (Wal dvogel et al 1999) method is similar to Flat Table method but the keys are calculated without group controller’s intervention. Mohamed and Kara (2003) have proposed a secret key multiplication for a scalable group rekeying protocol, without encryption and Decryption processes. The calculation procedure for rekeying is the same as that of SKM method. It is however shown that the later protocol performed better in terms of security and computational time.

Sudha et al (2009) have proposed a Secret Key Multiplication protocol (SKMP) that is based on modular Polynomial arithmetic over Galoic Field GF ($2^n$). The keys are transmitted in the form of modular polynomial over GF ($2^n$). Here the group is divided into small sub groups. The members of the sub group own an individual key $K_i$. Each sub group owns an auxiliary key which is shared by members of the subgroup. Group moderator owns a private key of the group. For rekeying, Group moderator has to change his private key from $K_c$ to $K_c'$. Auxiliary keys are then recalculated with the existing group member’s key, after every insertion or deletion processes. By this way of using simple multiplication and modular arithmetic rekeys are calculated.

Venkatesulu M and Kartheeban K. (2010) have proposed an euclidian algorithm based key computation protocol for group communication in a dynamic grid Environment. Here, each member of the group is assigned with a Personal Security Number (PSN). At the time of its embedment with member’s certificate, the Key Distribution Center (KDC) selects a group key $GK$ as a random value. The quotient and remainder of division operation is calculated with respect to member’s personal identity as shown in Equation (4.70), (4.71).

$$Q_i = \frac{GK}{PSN_i} \quad (4.70)$$
R_i = GK \mod \text{PSN}_i \tag{4.71}

KDC announces the pair \((Q_i, R_i)\) to all members of the group. Respective members can then calculate \(K_i\) by using Equation (4.72) and its security standard is also discussed. It is also proved that determining the value \(GK\) is very difficult.

\[ K_i = Q_i \cdot \text{PSN}_i + R_i \tag{4.72} \]

Later Venkatesulu and Kartheeban (2011) have enhanced the earlier protocol with avoidable measures. In this protocol, each member of a group is assigned with a large Prime \(q_i\) with the constraint that Prime Numbers must be greater than the value of common key \(K\). KDC can generate a message \(\text{Msg}\) by multiplying the contribution of the members and add the common key \(K\) by using the Equation (4.73) which is given below.

\[ \text{Msg} = (q_1 \cdot q_2 \cdot q_3 \ldots) + K \tag{4.73} \]

The KDC will multicast it to its members. After receiving it the members can derive the key by applying modulus operation by its own group secret key as given in the Equation (4.74).

\[ K = \text{M} \mod \text{P}_i \text{ for all } i \tag{4.74} \]

For every member join, KDC calculates the key by including the new member’s contribution as stated in the Equation (4.75) given below.

\[ \text{Msg}' = ((q_1 \cdot q_2 \cdot q_3 \ldots \cdot q_n) \cdot q_{n+1}) + K \tag{4.75} \]
It sends the value to all the members. From this message, each member can calculate the group key, just by taking the modular operation on the message \( \text{Msg.} \) by its respective contribution (Prime Number).

For member leave, KDC selects a new \( K' \) in such a way that \( K' < q_i \) (for all \( i \)) and calculate \( \text{Mag}' \) by simply skipping the contribution of the left member.

Group controller multicast the value of \( M' \) to all members. Each member can then calculate his own Group Key by simply take mod on its own identity premium. Hence, it is proved that it is an efficient, certificate free key computation protocol for securing group communication.

If more than one member is changed, then the key is computed so as to synchronize or to maintain consistency of partner’s keys of the group. Thus it is pondered to think for the best hybrid approach. Here, the contribution of the members is taken for calculating the key. But the key is calculated by the group controller not in distributed manner.

The responsibility of key calculation and distribution lies on the Individual administrator of the group. The robustness is assured by the freshness of the member’s contributions used to calculate the key. The weightage value for both the authentication mechanism have been arrived at based on the above discussion and is presented in Table 4.5.

### Table 4.5 Empirical weightage values for various levels of group key based authentication

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Authentication Mechanism</th>
<th>Level</th>
<th>Weight ages on Total Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Usability</td>
</tr>
<tr>
<td>1</td>
<td>Centralized Group Key Management protocol</td>
<td>Level 1*</td>
<td>0.6</td>
</tr>
<tr>
<td>2</td>
<td>Distributed Group Key Management</td>
<td>Level 2**</td>
<td>1.2</td>
</tr>
</tbody>
</table>

*Usability reduced by 20% of simple password (Table 4.2)

** Presuming 100% increase of usability from Level 1
One of the main drawbacks identified in the above discussed protocol is, that the rekeying procedure is more or less common for both the member join and member leave operations. Here the code for rekey operation is executed for every change in the membership.

Member join operation can be considered as an event intentionally done. The member leave may happen because of any transient nature of internet. Thus rekey operation for leave operation should not be the same as that of member joins operation. Leave operation may not happen intentionally. Thus a separate rekey mechanism has to be designed distinctly for leave operation. Also the rekey mechanism has to preserve or maintain Perfect Forward Secrecy in case of member leave operation and Perfect Backward Secrecy in case of member join operation.

### 4.7 INFERENCE FROM WEIGHTAGE CALCULATIONS

Based on the social nature with respect to the weightages on the three parameters taken for the present research, the optimum valued security mechanisms have been considered for the design of the proposed model. The chosen mechanisms have been highlighted in the respective table. Considering these mechanisms, models have been proposed through algorithms that have been designed by the researcher. These algorithms have been coded and implemented in a selected SNS (chokut.com) and experimented before validating through a social survey. The procedure is explained in the next chapter. The summary of merits and demerits of authentication mechanisms as discussed by O’Gorman (2003) are presented Table 4.6.
Table 4.6 Merits and demerits of various authentication mechanisms

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Authentication Mechanism</th>
<th>Merits</th>
<th>Demerits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Password based</td>
<td>Easy to remember, traditional way of authentication</td>
<td>Vulnerable to Dictionary attack and Brute force attack</td>
</tr>
<tr>
<td>2</td>
<td>Token based</td>
<td>Ensure the validity with availability of Smart card /RFID Tag</td>
<td>It is mandatory to hold the card/tag at all time and in all places</td>
</tr>
<tr>
<td>3</td>
<td>Bio Metric based</td>
<td>Advanced way of Authentication. It will be considered as a highly secured and fault free way to ensure the person’s Identity.</td>
<td>Not supports non human entity. It is mandatory to have huge database to store the biometric templates at the verification end. It is mandatory to have a special device to submit the response</td>
</tr>
<tr>
<td>4</td>
<td>Personalized Prompt</td>
<td>Remedial measure to overcome the impersonation attack.</td>
<td>Strength of the security purely depends on hardness of the question</td>
</tr>
<tr>
<td>5</td>
<td>Personal Interview</td>
<td>When the matching factor of biometric response and template get deviated (example: changes in the human voice due to whether, Facial response differs according to the mood of the person). It can support an additional mechanism to prove the Identity of the user.</td>
<td>Vulnerable through third party /customer support.</td>
</tr>
<tr>
<td>6</td>
<td>Authentication associated with user device Identity</td>
<td>Ensure the user identity with the help of the user’s device address</td>
<td>It will not support the pervasive environment. It indirectly forces the users to hold the device all the time. If the device is not portable in nature then this mechanism will not support.</td>
</tr>
</tbody>
</table>
A total empirical weightage value of more than 1.5 is taken for experimentation from all the above analytical results. Corresponding security mechanisms have been taken for further study through experimentation.

Based on the above findings, the researcher has attempted to propose a reliable authentication model for SNS. This model has been designed based on the above levels implemented, and validated. They are elaborated in the next chapter.