Abstract: - In a digital signature scheme, the security of the private key is of vital importance. If the private key is ever compromised, it can be used to sign forge documents. The conventional method of secure private key storage is through password-based encryption. However, user-chosen passwords have very low entropy, which may be exploited by an attacker to launch password-guessing attacks. In order to increase the security of a digital signature scheme, we propose a novel method that uses fingerprint, password and smart card to dynamically generate a private key for digital signatures. The scheme is sufficiently robust to generate a constant key and is tolerant to errors generated in the fingerprint pattern at the time of key generation. Our proposed algorithm is capable of generating key lengths that can meet the current security requirements of public key algorithms used for digital signatures and is more secure than traditional password-based method of protecting a private key.

Key-Words: - Biometrics, Cryptography, Image processing

1 Introduction

In a public key cryptosystem, a message is signed with the private key and is verified with the corresponding public key. The security of a private key is therefore very important. If the private key is ever compromised, extensive damage can be caused to the user. The key can be used by an attacker to sign forge documents or to decrypt secret messages. Because of the large size of a cryptographically-strong key, it is not feasible for a user to remember the private key and enter each time it is required. Instead, the private key is usually encrypted with a user chosen password. To retrieve the key, the user will have to enter the same password in order to perform successful decryption of the key. A common problem with password-based method is the low entropy, which may be exploited by an attacker to launch password guessing attacks [1]. The space of passwords is very limited. For an eight-character password, there are approximately $2 \times 10^{14}$ combinations of English alphabets, both upper and lower case and digits. This is very small as compared to the size of a 1024-bit RSA key. Moreover, using a low-entropy password as a key to a strong cryptographic algorithm can transform it into a weak one. Therefore efforts are required to devise new methods that can increase the level of security the current password-based techniques offers in the safe custody of the private key.

In order to make key storage mechanism more secure, researchers are now looking into the use of biometrics. A biometric is a person’s unique physical or behavioural characteristics that can be used to identify an individual. Physical characteristics includes fingerprints, hand geometry, retina, iris and facial characteristics, etc. Behavioural characteristics includes signature, voice, keystroke patterns and gait, etc. [2]. The basic aim of using
biometrics is to devise a mechanism that is more secure in protecting the cryptographic key of a user as compared to conventional method of password-based encryption. One of the primary limitations with biometrics is that they can never provide absolutely certain measurements because personal features have a natural range of variation. This makes the task of linking biometrics with cryptographic algorithms quite challenging. Although password-based systems are weak from security perspective, they are 100% reliable. The presentation of a correct (incorrect) password will always correctly result in acceptance (denial) of a service. In this paper, we propose a novel method that combines password with fingerprint and information stored in a smart card to dynamically generate a private key that can be used for generating digital signatures. We will show that our proposed scheme is much more secure then conventional methods of storing a key through password-based encryption.

2 Prior Work
The notion of using biometric template directly as a cryptographic key was first proposed by Bodo in a German patent [3]. In this method, the data derived from the biometrics is used directly as a cryptographic key. This method suffers from two main limitations. First, since a biometric template is not consistent due to environmental and physiological factors, this may not be able to generate a constant value of the key every time. Secondly, if the key is ever compromised, then the use of that biometric is irrevocably lost. Therefore this technique cannot be realized in a real world system especially where periodic updating of the cryptographic key is required. Another innovative idea of generating a cryptographic key from voice characteristics of a user is proposed by Monrose et al. [4]. Using this technique, a 46-bit key can be generated from a roughly two second spoken password. The idea of using online handwritten signatures for private key generation has recently been proposed [5]. The scheme can generate a key of 160-bits. In another method that uses fingerprints [6,7], the key is linked with the biometric at the time of enrolment to form a data that gives no information of the key or the biometric. In the verification stage, the key is then reconstructed using the stored data and the biometric. The paper illustrates an example of generating a 128-bit key that can be used for cryptographic applications.

3 Our Method
All the methods discussed above have a limitation of key length. For example, the largest key size that can be produced is 160-bit with the technique presented by Feng et al. [5]. However in public key algorithms like the RSA, key lengths of 1024-bit or higher are required [8], keeping in view the current computational power. In order to solve this problem, we propose a novel technique in this paper that besides generating keys of higher lengths also improves the level of security the current password-based scheme offers for protection of a private key. Rather than storing a private key by password encryption, our method combines password, fingerprint features and data stored in a smart card along with some login information to dynamically generate a private key on demand. Once the key is generated, it can be used with the specific public key algorithm to generate digital signatures. The proposed scheme uses minutia points for image alignment [9] and the orientation field of a ridge map of a fingerprint as a feature for key generation. This feature is designated as the Ridge angle vector, \( R_a \) and is calculated using the algorithm proposed in [10]. The minutia points are extracted using the algorithm proposed in [9]. The orientation field of the original fingerprint image does not give a very accurate measurement of the ridge flow because of the presence of noise and broken ridge lines. This is shown in Fig.1 and Fig. 2.
However, if we calculate the orientation field of the ridge map, a reasonably accurate estimation of the ridge flow is obtained. The result is shown in Fig. 3 and Fig. 4. As compared with Fig. 2, the orientation field in Fig. 4 now follows the ridges very smoothly and can be used as a feature for key generation process. We are restricted to use the minutia points as features for key generation because of three main reasons. Since image alignment requires storage of template minutia, this may enable an attacker to generate the key if the template is ever compromised. Secondly, the number of minutia points are not same among different people. In addition, there are a number of spurious minutia detected, which makes the key generation task extremely unreliable. The proposed scheme consists of an Enrolment Phase and Signature Generation Phase. We denote a legal user by $U_a$, his fingerprint by $f_a$, the minutia points extracted by $M_a$, password by $pwd_a$, the smart card by $Scard_a$ and the private key by $Kp_a$. We assume that the user already has his public/private key pair, as the basic purpose of our algorithm is to link it with the existing private key of a user. In case if a new key pair is required, it can be generated by any public key algorithm like the RSA, etc.

3.1 Enrolment Phase

In this phase, $U_a$ submits $f_a$, $pwd_a$, $Scard_a$ and $Kp_a$ to the system. The system then does the following:

1. Compute $M_a$ and $R_a$ from $f_a$, where $R_a = \{ \theta_m : 1 \leq \theta_m \leq 180 \text{ degrees and } 1 \leq m \leq 180 \}$. Store $M_a$ in $Fmin_a$.

2. For each $\theta_m \in R_a$, define $\theta_{sk}$, where $S = m = 180$ and form groups $G_i (1 \leq i \leq 36)$ comprising of five consecutive $\theta_{sk}$. The variable $r$ is the system parameter defining the number of shadow angles. The purpose
of shadow angles is to make the system tolerant to errors in the fingerprint pattern at the time of key generation.

\[ \theta_{sk} = \theta_a, \theta_m, \ldots \pm t_{1(2r+1)}, 1 \leq k \leq (2r+1) \] (1)

3. Define a set \( P \) containing the \( P_i \) segments of \( Kp_a \), i.e. \( P = \{P_1, P_2, \ldots, P_{36}\} \).

4. Each consecutive \( P_i \in P \) is treated as a secret and an \((m-n)\) threshold scheme [11] is used to generate the secret shares, where \( m \) is the minimum number of shares required to retrieve the secret and \( n \) is the total number of shares. For our experiment, \( m = 2 \) and \( n = 5 \times (2r+1) \).

5. Compute random, collision free hashes \( H'_{sk} \) using equations (2) and (3). The function \( H_{SHA} \) in equation (2) uses SHA-1 algorithm [12] to give 160-bit random and unique hashes, \( \zeta \) is a public value selected arbitrary for every \( S \). Equation (3) maps the 160-bit hashes generated by equation (2) into a small range of \( \theta_{sk} \). The parameter \( \sigma \) defines the maximum range of \( H'_{sk} \) and \( \delta \) is a random seed value chosen for every \( S \) in order to avoid any collision in the values of \( H'_{sk} \).

\[ H_{sk} = H_{SHA}(\theta_{sk}, pwd_a + \theta_a, psw) \] (2)

\[ H'_{sk} = f_0(H_{sk}, \delta, \sigma) \] (3)

6. For each segment of \( Kp_a \) store the secret shares generated in Step 4 in \( RandomSecret_a \) by the corresponding index pointed by \( H'_{sk} \). For example, for the \( n \) shares of \( P_i \), the first \( 5 \times (2r+1) \) \( H'_{sk} \) values will be used where, \( 1 \leq S \leq 5 \), and \( 1 \leq k \leq (2r+1) \).

7. Encrypt \( \delta \) values with \( psw_a \) and store them in \( Scard_a \). Encrypt \( Fmin_a \) and \( RandomSecret_a \) with \( psw_a \) to get \( EFmin_a, ERandomSecret_a \) respectively and store them in any medium like hard disk, etc. We call \( EFmin_a \) and \( ERandomSecret_a \) as the registry files for user \( U_a \).

**3.2 Signature Generation Phase**

In this phase, the system will generate the private key if and only if the fingerprint, password and smart card are provided by the legitimate user. The private key can then be used with the corresponding public key algorithm to generate digital signatures. The working of this phase is as follows: \( U_a \) submits \( f_a, psw_a, Scard_a \) to the system. The system reads the corresponding registry files \( EFmin_a, ERandomSecret_a \) and does the following:

1. Decrypts \( EFmin_a \) using \( psw_a \) to get \( M_a \) and use a minutia alignment algorithm [9] to align \( f_a \) at a position presented in the enrolment phase.

2. Compute \( R_a \) from \( f_a \), where \( R_a = \{\theta_m : 1 \leq \theta_m \leq 180 \) degrees and \( 1 \leq m \leq 180 \} \). For each \( \theta_m \in R_a \) define \( \theta_{vk} \), where \( V = m = 180 \) and form Groups \( G_i \) (\( 1 \leq i \leq 36 \)) comprising of five consecutive \( \theta_{vk} \).

\[ \theta_{vk} = |\theta_m - 1, \theta_m, \theta_m + 1|, 1 \leq k \leq 3 \] (4)

3. Read the encrypted \( \delta \) values from \( Scard_a \), decrypts them with \( psw_a \) and calculate \( H'_{vk} \) using equations (5) and (6)

\[ H_{vk} = H_{SHA}(\theta_{vk} + H_{SHA}(pwd_a + \zeta)) \] (5)

\[ H'_{vk} = f_0(H_{vk}, \delta, \sigma) \] (6)

4. Decrypt \( ERandomSecret_a \) with \( psw_a \) and read the secret shares based on the index pointed by \( H'_{vk} \). Compute the secrets i.e. the segments of \( Kp_a \) using polynomial interpolation [13]. If any two secrets calculated within a group matches, the secret i.e. the segment of the private key is considered valid. The concatenation of the segments \( P_i \) to \( P_{36} \) will form the private key for \( U_a \). Once the private key is generated, it can be used with the corresponding public key algorithm for generating digital signatures.

**4 Security Analysis**

In this section, we discuss the security of our proposed technique. We show that the cost of finding the correct private key is greater by a significant multiplicative factor as compared to a simple password-based scheme. We describe two types of attacks against our scheme, on-line attack
and off-line attack. In an on-line attack, the attacker has to produce the legitimate user password, fingerprint and smart card to dynamically generate the private key for digital signature. This is an obvious improvement as compared to the traditional method in which only a password is required to decrypt the private key for digital signature. In an off-line attack, we assume the security of the smart card and analyze our scheme against an attack on the registry files $EF_{min}$ and $E_{RandomSecret}$. We assume that the attacker has managed to capture these files and in addition is also successful in guessing the correct password to get the decrypted version $F_{min}$ and $RandomSecret$. $F_{min}$ only contains user minutia points and will not reveal any information regarding $Kp$. Using $RandomSecret$ even if the attacker is able to construct all secrets through the $(m-n)$ threshold scheme [11], he will still not be able to get the exact sequence that makes up the private key. For this, he will have to perform at most $36!$ permutations, i.e. approximately $3.7 \times 10^{41}$ number of iterations and in each iteration check for key validity. This is a considerable high overhead as compared to an eight-character password space, which has roughly $2 \times 10^{14}$ different combinations of English alphabets both upper and lower case and digits. Even if a sixteen-character password is chosen, the total possibilities are roughly $5 \times 10^{36}$ which is still low as compared to $3.7 \times 10^{41}$. In addition, since the file $RandomSecret$ contains secret shares, it will not reveal any information of the ridge angle features i.e. the biometric used for the key generation process.

5 Condition for Successful Key Generation

Assume we have a genuine user ‘a’ whose fingerprint sample is represented by $f_a$ and password by pwd$_a$ at the time of enrolment. Let $R_a$ be the ridge angle vector of $f_a$. In the enrolment phase, for each $\theta_{i} \in R_a$, a set $\theta_{Sk}$ is formed given by equation (1). The number of elements in the set $\theta_{Sk}$ depends on the value of the shadow angle ‘r’. As discussed in the enrolment phase, a total of 36 groups ($Ga_i$, where $1 \leq i \leq 36$) will be formed where each $Ga_i$ comprises of five consecutive $\theta_{Sk}$. The system will then calculate the hashes $H_{Sk}$, $H_{Sk}^*$ and will store the random seed values $\delta$ in the smart card ($Scard_a$) of the user. Now suppose at the time of key generation the same user presents his fingerprint which we represent as $f_a^*$, password (pwd$_a^*$) and smart card ($Scard_a$). Let $R_a^*$ be the ridge angle vector for $f_a^*$. In the key generation phase, for each $\theta_{i}^* \in R_a^*$, a set $\theta_{V_k}$ is formed given by equation (4). Similarly a total of 36 groups ($Ga_i^*$, where $1 \leq i \leq 36$) will be formed where each $Ga_i^*$ comprises of five consecutive $\theta_{V_k}$. In order for each $Ga_i^*$, to match with $Ga_i$, it is necessary that at least any two $\theta_{V_k}$ should match with any two $\theta_{Sk}$ in the respective groups. Further, for any $\theta_{V_k}$ to match with $\theta_{Sk}$, it is necessary that at least two elements of the respective $\theta_{Sk} \in \theta_{Sk}$ (for $V=S$). The private key of user ‘a’ will only be generated successfully if the following conditions are satisfied:

a. For all $i=j$, $Ga_i$ should match with $Ga_j^*$.

b. If condition(a) is satisfied and $pwd_a^* = pwd_a \Rightarrow H_{V_k} \in H_{Sk}$ (for $V=S$).

c. If conditions (a and b) are satisfied and $Scard_a^* = Scard_a \Rightarrow H_{V_k}^* \in H_{Sk}^*$ (for $V=S$).

6 Key Length

Our proposed algorithm is capable of generating keys of any length. This can be achieved through Step 3 and 4 in the enrolment phase. The basic idea is to divide the key into 36 segments and treat each segment as a secret. Table 1 shows the size of the secret for different key lengths obtained by dividing the total key length by 36 and rounding-off the result to the nearest integer. The extra bits introduced due to rounding-off are padded with random data bits that are removed in the key generation phase.

<table>
<thead>
<tr>
<th>Key Length (bit)</th>
<th>Size of Secret</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>15-bit</td>
</tr>
<tr>
<td>1024</td>
<td>29-bit</td>
</tr>
<tr>
<td>2048</td>
<td>57-bit</td>
</tr>
<tr>
<td>4096</td>
<td>114-bit</td>
</tr>
</tbody>
</table>
7 Genuine Acceptance and Imposter Rejection

The novelty in our design is that it keeps the genuine acceptance high while rejecting an imposter. The genuine acceptance can be increased by increasing the shadow angles in the enrolment phase. This would enable a constant value of the key to be generated for a legitimate user and will be tolerant to errors in the fingerprint pattern. Fig. 5 shows one of our implementation in which 270 genuine fingerprint comparisons were made for different shadow angles. We were able to get a 100% genuine acceptance at $r=10$ for all the fingerprint samples. However, increasing the shadow angles will not increase the false acceptance because the hashes produced by equations (2) and (3) not only depends on the fingerprint data but also on the password and random seed values. Therefore even if two different individuals have fingerprints that are quite similar, our algorithm can discriminate them very sharply since the indexing of the secret shares that forms the private key is a function of password, fingerprint and random seed values.

![Fig.5: Genuine Acceptance Rate](image)

8 Conclusion

Despite the fact that public key algorithms used for digital signatures are very strong, their security lies in the fundamental assumption that the private key is kept safe. For an eight-character password, there are approximately $2 \times 10^{14}$ combinations of English alphabets, both upper and lower case and digits. This is very small as compared to the size of a 1024-bit RSA key. In order to make the key storage mechanism more strong, we have presented a scheme that provides more security to digital signature generation by using a password, fingerprint and smart card. Our method enables the private key to be generated dynamically whenever a message is required to be signed with a digital signature. The algorithm is sufficiently robust to generate a constant private key and is tolerant to errors generated in the fingerprint pattern at the time of signature generation. The security analysis shows that our scheme is much more secure as compared to traditional method of protecting a private key using password-based encryption. Our proposed scheme offers a distributed security approach. Instead of encrypting a private key with a password and storing it at one place, we have proposed an idea that requires a valid smart card, correct password and fingerprint to generate the key every time it is required for digital signature. In addition, our proposed scheme can be used with any public key cryptosystem for digital signature generation as it can produce a key of any desired length.

References:


[5] Hao Feng, Chan Choong Wah, Private Key Generation from On-Line


