SECURITY ANALYSIS FOR KEY GENERATION SYSTEMS USING FACE IMAGES

Wende Zhang, Cha Zhang and Tsuhan Chen

Dept. of Electrical and Computer Engineering, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{wendez, czhang, tsuhan}@andrew.cmu.edu

ABSTRACT

In this paper, we analyze the security problem of user information associated key generation (UIAKG) systems. We consider three kinds of attacks from the hacker: the exhaustive search attack, the authentic key statistics attack and the device key statistics attack. Under each attack, we give the estimate of the number of guesses the hacker has to make in order to access the system. Such analysis provides useful guidelines in designing a UIAKG system. The analysis also suggests that a user-dependent UIAKG is more secure than a user-independent one. Two UIAKG systems using face images as inputs are designed and compared to support the above theoretical analysis.

1. INTRODUCTION

Security applications often need certain private information to authenticate the user's privilege. Digital keys are widely used to serve such a purpose in many applications. For example, we use a PIN number as the key to access ATM accounts; we use a password as the key to login the computer system; we also use keys for data encryption / decryption.

Given certain input from the user, a traditional key generator often generates a long key at random [1][2]. Such strategy can prevent the hackers' exhaustive search attack as the entire key space is very large. However, a long key is easily forgotten and not user-friendly. Hence it has a high false reject rate (FRR).

Instead of asking users to memorize long keys, some recent key generation systems [3]-[10] have been trying to generate keys based on the users' personal information, e.g., the users' biometric information. We call such systems *user information associated key generation (UIAKG) systems*. Compared with the traditional key generator, UIAKG systems are often designed based on a close set of subjects. Such set of subjects form the *subject space* of the key generation system. The keys generated by UIAKG systems are thus limited by the variation in the subject space, and cannot be both distinguishable and arbitrarily long. It is therefore very important to know how secure UIAKG systems really are under the above constraints.

Given the subject space, a UIAKG system can be either *user-independent* [4][5][6] or *user-dependent* [7][8]. A user-independent UIAKG system creates a single piece of device, which can be used to generate keys for all subjects in the subject space. This is similar to the traditional key generation systems, except that the subject space is a close set. A user-dependent UIAKG system, in contrast, will create many devices, one for each subject. A subject can only access the device created specifically for the same subject, and will be denied if he/she wants to access a device belonging to another subject. In general, since the devices created by a user-dependent UIAKG system can be fine-tuned for their authentic subjects, they tend to have better authentication performance (in terms of false accept rate (FAR) and false reject rate (FRR)) than the devices created by user-independent UIAKG systems. On the other hand, although there has not been a proof which shows that user-independent UIAKG systems are more secure than user-dependent UIAKG systems, there is a wide concern that a user-dependent UIAKG system might expose the user's identity in an inexplicit way and is thus insecure.

In this paper, we present a comprehensive analysis on the security of general UIAKG systems. We classify the hacker's attacks into several categories, and show how robust a UIAKG system is under these attacks. Such analysis also serves as guidelines when designing a UIAKG system. Meanwhile, our analysis suggests that a user-dependent UIAKG system can be secure if it is carefully designed. Experimental results also demonstrate that user-dependent UIAKG systems outperform user-independent systems.

The paper is organized as follows. In Section 2, we briefly describe two biometrics-based key generation systems, one being user-independent and the other being user-dependent. In Section 3, we detail the security analysis for general UIAKG systems. Experimental results are shown in Section 4 and conclusions are given in Section 5.

2. TWO TYPICAL UIAKG SYSTEMS

In this section, we introduce two UIAKG systems in the application of key generation from subject face images. Note that the detailed algorithms described below can be replaced by other biometrics such as fingerprints or iris images, and by other user inputs such as passwords or passphrases. However, the general frameworks of the two systems are applicable to many other user inputs.

The first system is user-independent. It creates a single device for generating keys for all the subjects, as shown in Figure 1. The biometric features are first extracted from the biometric data using the feature extraction module. For example, we use Principal Component Analysis (PCA) [11] to extract the eigen-coefficients from the data as the biometric features. A user-independent key generator then produces the authentic keys

0-7803-8554-3/04/$20.00 ©2004 IEEE.

of the subjects based on the biometric features. In this paper, we binarize each feature into 1 bit by a user-independent threshold, which is the global mean of the feature values for all the subjects. Some of these features are selected as distinguishing features based on the separation between the authentic and imposter values. A secret sharing method [5] is then used to test whether all the bits of the input biometrics matches with the authentic bits, which are computed from the distinguishing features. This process resembles the algorithm used in [5].

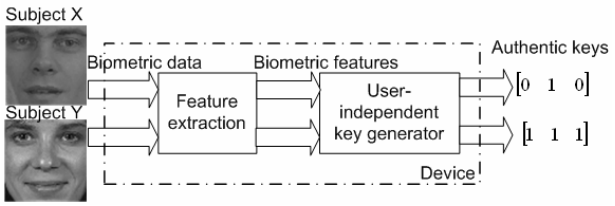

Figure 1 A user-independent key generation system.

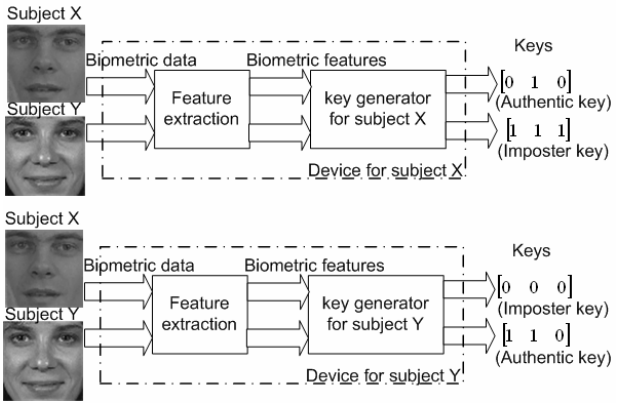

Figure 2 A user-dependent key generation system.

Figure 2 shows a user-dependent UIAKG system. The system creates different devices for different subjects. For example, the device for subject *X* contains a key generator specifically designed for him. When subject *X* tries to access the device, the device will generate an authentic key and grant the access. If subject *Y* also tries to access the device, the device will generate an invalid key, which denies the access. In this paper, we will refer to such invalid keys as the imposter keys.

To design such a system, we first extract biometric features using the feature extraction module, which can be shared by all the devices. Then the user-dependent key generators are designed based on the biometric features. In this paper, we design the key generators such that each feature is binarized into 1-bit by a user-dependent threshold to minimizing the authentication error rate (FAR and FRR). Interested readers are referred to [8] for more details about the algorithm.

3. SECURITY ANALYSIS FOR UIAKG SYSTEMS

The hackers have many ways to break into a key generation system. In this section, we examine the average number of guesses they have to make in order to achieve the authentic key using different attacking methods. Such analysis also serves as a guideline for designing a secure UIAKG system (e.g., a system that denies the access after the failure of a certain number of trials).

3.1. Exhaustive Search Attack (ESA)

If the hacker does not have any information about the subject space or key statistics information, he/she has to perform an exhaustive search in the entire key space. Let the key length be L. If the key space is very large, i.e., L is very large, the expected number of guesses by exhaustive search is roughly:

$$N \approx 2^{L-1} \tag{1}$$

Therefore, a longer key is more secure under ESA. This is a well-known result.

3.2. Authentic Key Statistics Attack (AKSA)

If the hacker knows the statistics of the authentic keys generated by the key generator system, he/she may try to guess the keys smartly. Let $s_i, i = 1,2,\cdots,M$ be the M subjects considered during the design of the key generation system. Let $K_i, i = 1,2,\cdots,M$ be their corresponding authentic keys. Since it is possible for a key generation system to generate the same authentic key for different subjects, we denote $K'_j, j = 1,2,\cdots,P$ as the keys among $K_i, i = 1,2,\cdots,M$ that are unique and ordered by their frequency $f_j, j = 1,2,\cdots,P$. That is, we assume $f_1 \geq f_2 \geq \cdots \geq f_P$. The estimated number of guesses the hacker must make is thus:

$$N_{AKSA} = \sum_{j=1}^{P} j \cdot f_j \tag{2}$$

Equation (2) assumes that the hacker makes the guess in the order of the authentic key frequency (from high to low), which is the best strategy he/she can use. Equation (2) is also used in [4] for security analysis, which is given the name *guessing entropy* [12][13].

In a user-independent key generation system, once the feature vectors and the thresholds are determined, the keys for all the subjects are determined. Therefore, in order to increase the guessing entropy, the system designer must select the feature vectors and thresholds very carefully. This is not an easy task, as tuning the feature vectors and thresholds may also affect the FAR and FRR of the system. The system designer usually has to make a tradeoff between the guessing entropy and the FAR/FRR.

In a user-dependent key generation system, such problem does not exist, thanks to the different feature vectors and thresholds one may use for different subjects. The system designer can easily select different authentic keys for different subjects, because each subject has his/her own feature space, which can be binarized per the system designer's wish independently. As long as the size of the entire key space is larger than the number of subjects, i.e., $2^L > M$, a user-dependent system designer can always make sure that $K_i, i = 1,2,\cdots,M$ are all different from each other, resulting the largest possible guessing entropy:

$$N_{AKSA} = \sum_{i=1}^{M} i \cdot \frac{1}{M} = \frac{M+1}{2} \tag{3}$$

3.3. Device Key Statistics Attack (DKSA)

If the hacker knows the subject space of the system, he/she may probe the key generation system by inputting the subject information and collecting the statistics of the generated keys.

Given such statistics, the hacker may have better ways to guess the authentic key. Since such attack focuses on a single piece of device, we name it the device key statistics attack (DKSA).

In a user-independent key generation system, there is only one single device. When the hacker inputs a subject to the device, he/she will always get an authentic key. Therefore, the DKSA for a user-independent key generation system is equivalent to its AKSA.

However, in a user-dependent key generation system, the DKSA is no longer the same as the AKSA. Given a device which belongs to a single subject, only the subject who owns the device will result in an authentic key. The remaining subjects will get some imposter keys which might be different from their own authentic ones.

By probing the device with all the subjects, the hacker can collect a set of unique keys, denoted as $K'_j, j=1,2,\cdots,P$. We may order these keys by their frequency $f_j, j=1,2,\cdots,P$ in non-decreasing order, i.e., $f_1 \le f_2 \le \cdots \le f_P$. The problem is, following which order the hacker should try these keys.

A UIAKG system often needs to improve its authentication performance by reducing FAR and FRR. However, such property can also be utilized by the hacker to attack the system. For instance, if the hacker knows that the system's FAR is very low, he/she will try the keys in the order $K'_1, K'_2, \cdots, K'_P$. This is because a low FAR of the device implies that the authentic subject will not share his/her key with many other subjects. Therefore, trying the key that has the least frequency will have the largest chance of success.

We next present a scheme to calculate what is the number of expected guesses for a user-dependent system under DKSA. Consider a user-dependent system which has M subjects $s_i, i=1,2,\cdots,M$. For subject s_i, we can build a device for him/her. Let $K'_{ij}, j=1,2,\cdots,P_i$ be the unique keys generated when inputting all the subjects to the device and $f_{ij}, j=1,2,\cdots,P_i$ be the frequencies of the keys. Again K'_{ij} is ordered in non-decreasing frequency order, i.e., $f_{i1} \le f_{i2} \le \cdots \le f_{iP_i}$. Among these P_i unique keys, one of them is the authentic key. Let it be $K'_{iq}, 1 \le q \le P_i$. We say that an event E_q has happened when the q^{th} unique key K'_{iq} is the authentic one for the subject s_i.

When all the subjects are considered, we may count the frequencies of $E_q, 1 \le q \le \max_i P_i$. Let the frequencies be $F_q, 1 \le q \le \max_i P_i$. The best strategy of the hacker is thus to assume event E_q has happened in a decreasing order of F_q. That is, if E_q happens most frequently for all the subjects, the hacker will try the q^{th} unique key K'_q first. The expected number of guesses is thus:

$$N_{DKSA} = \sum_{j=1}^{\max_i P_i} j \cdot F'_j \qquad (4)$$

where $F'_j, 1 \le j \le \max_i P_i$ is the reordering of $F_q, 1 \le q \le \max_i P_i$ in non-increasing manner.

Table 1 shows a very simple user dependent key generation system. It has 6 subjects, and the key space is 2 bits. The authentic keys are along the diagonal direction (marked with bold font). Among the authentic keys, 00 and 11 appear once, 01 and 10 appear twice. Therefore, under AKSA attack, the guessing entropy of the system is:

$$N_{AKSA} = 1 \times \frac{2}{6} + 2 \times \frac{2}{6} + 3 \times \frac{1}{6} + 4 \times \frac{1}{6} = 2\frac{1}{6} \qquad (5)$$

Table 1 A simple user-dependent key generation system

Subject / Device	s_1	s_2	s_3	s_4	s_5	s_6
For s_1	**00**	01	10	01	10	10
For s_2	10	**01**	11	11	11	10
For s_3	00	00	**10**	11	11	11
For s_4	00	00	00	**11**	01	01
For s_5	01	01	11	01	**10**	11
For s_6	11	10	11	10	10	**01**

Under DKSA attack, consider the device for subject s_1 as an example. There are three unique keys: 00, 01 and 10. Key 00 is the authentic key, which appears once. 01 appears twice and 10 appears three times. If we order the unique keys based on their frequency, the order should be $K'_{11}=00$, $K'_{12}=01$ and $K'_{13}=10$, with frequency $f_{11}=\frac{1}{6}$, $f_{12}=\frac{2}{6}$ and $f_{13}=\frac{3}{6}$. Since K'_{11} is the authentic key, event E_1 has happened for the subject s_1. Similarly, we may perform the same procedure for the other subjects. When all the subjects are considered, it is not difficulty to see that only E_1 happens 6 times, with frequency 1. E_2 and E_3 never happen. Therefore, when the hacker tries to access a device, he/she will always guess the authentic key be the first unique key in non-decreasing frequency order. By doing so, the hacker will obtain the authentic key in one shot. The number of guesses is thus:

$$N_{DKSA} = 1 \times 1 = 1 . \qquad (6)$$

Therefore, this system is very vulnerable to DKSA attack.

Knowing how AKSA and DKSA are performed, we should be careful when designing a user-dependent key generation system. We first need to make sure that different subjects should have different authentic keys. This is easily achievable by manipulate the binarization process for each device. Furthermore, to avoid DKSA, event $E_q, 1 \le q \le \max_i P_i$ should better be uniformly distributed with respect to q. An easy way to guarantee this is to let the keys (including the authentic key and the imposter keys) generated for different subjects be different from each other in every created user-dependent device. The hacker will then not be able to know which key to use when starting the guessing process and have to try them all one by one. Such goal is not difficult to achieve, because usually the key space is much larger than the subject space. A large key space will very likely make all the subjects generate different keys.

4. EXPERIMENTAL RESULTS

Experiments are conducted on the AMP face database with expression and registration error (as shown in Figure 3) to compare

the performance of the user-dependent and user-independent UIAKG systems. We take 20 subjects in this database for evaluation. Each subject has 137 face images at size 64×64. We use 25 images of each user to train the feature extraction module. Principal Component Analysis is performed on all the training images to reduce the features' dimensionality. The first 100 eigen-coefficients are taken as the biometric features in Figure 1 and Figure 2. We use another 25 images of each user to determine the threshold for each feature. The remaining 87 images of each user are used as test images to evaluate *FAR*, *FRR* and the expected number of guesses the hacker need to make (*N*).

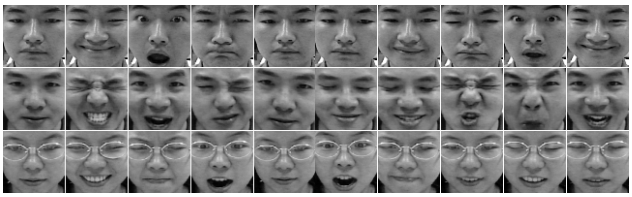

Figure 3. Sample images of AMP face database with expression and registration error.

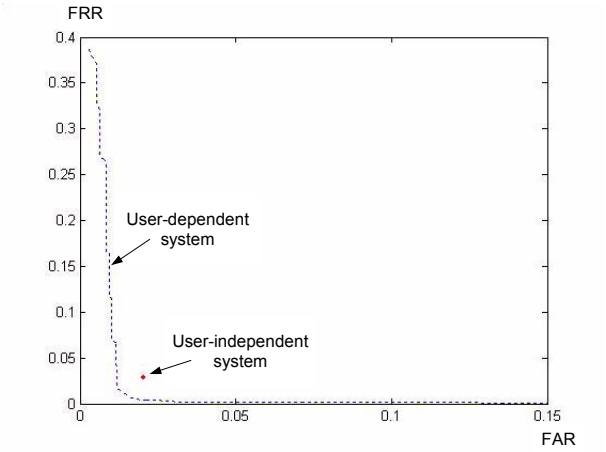

Figure 4 FAR and FRR performance of the two systems.

Figure 4 shows the FAR and FRR performance of the two systems when we fix the number of distinguishing features used in the system as $L = 5$ (therefore the key length is 5 bits). As the user-independent system has fixed thresholds for all the features, we only obtain one operating point, marked as a red dot in Figure 4. The dashed curve represents the user-dependent system's FAR and FRR. Obviously, the user-dependent system performs better, because at the same FRR, its FAR is lower, or at the same FAR, the FRR is lower. The operating point of the user-dependent system can be decided by the system designer based on the design specifications.

Table 2 Performance and security comparison of the two systems

	FAR	*FRR*	*L*	N_{AKSA}	N_{DKSA}
User-independent	2.0%	2.8%	5	2.9	2.9
User-dependent	1.4%	1.0%	5	10.5	4.38
User-independent	0%	53.4%	17	7.95	7.95
User-dependent	1.0%	0.9%	17	10.5	8.27

Table 2 shows the performance and security comparison of the two systems at different key lengths. In both configurations, the user-dependent system has much better authentication performance. Under AKSA, the number of guesses N_{AKSA} for the user-dependent system is always 10.5, which is the largest possible. Under DKSA, the user-dependent system also outperforms the user-independent system in terms of N_{DKSA}. Therefore, the user-dependent system is also more secure than the user-dependent system under hacker attacks.

5. CONCLUSIONS

We have given comprehensive analysis on the security of UIAKG systems under different kinds of attacks. We also showed that if a user-dependent system is designed carefully, it outperforms a user-independent system in both authentication performance and security. Such conclusions are supported by comparing the performance of two UIAKG systems using face images as inputs.

REFERENCES

[1] S. Burnett and S. Paine, *RSA Security's Official Guide to Cryptography*, RSA Press, McGraw-Hill/Osborne Media, 2001.

[2] A. Menezes, P. Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[3] C. Soutar, D. Roberge, A. Stoianov, R. Gilroy and B.V.K. Vijaya Kumar, "Biometric Encryption™," Chapter 22 in *ICSA Guide to Cryptography*, edited by R.K. Nicholls, 1999, pp.649-675.

[4] F. Monrose, M.K. Reiter, and S.G. Wetzel, "Password hardening based on keystroke dynamics," *International Journal on Information Security* 1(2), February 2002. pp. 69–83.

[5] F. Monrose, M.K. Reiter, Q. Li, and S. Wetzel, "Cryptographic key generation from voice," Proceedings of *the 2001 IEEE Symposium on Security and Privacy*, 2001, pp. 202-213.

[6] F. Monrose, M.K. Reiter, Q. Li and S. Wetzel, "Using voice to generate cryptographic keys," *Proceedings of 2001: A Speaker Odyssey, The Speaker Recognition Workshop*, June 2001, pp. 237–242.

[7] Y. Chang, W. Zhang and T. Chen, "Biometric-based cryptographic key generation," submitted to *IEEE Conference on Multimedia and Expo,* 2004.

[8] W. Zhang, Y. Chang and T. Chen, "Optimal thresolding for key generation based on biometrics," submitted to *IEEE Conference on Image Processing,* 2004.

[9] J. Daugman, "High confidence visual recognition of persons by a test of statistical independence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15(11), 1993, pp. 1148-1161.

[10] C. Ellison, C. Hall, R. Milbert, B.Schneier, "Protecting secret keys with personal entropy," *Future Generation Computer Systems*, 16, 2000, pp. 311-318.

[11] I.T. Jolliffe, *Principle Component Analysis*, Spring-Verlag, New York, 1986.

[12] J.L. Massey, "Guessing and entropy," *Proceedings of the 1994 IEEE International Symposium on Information Theory*, Trondheim, Morway, 1994, pp.204.

[13] C. Cachin, *Entropy measures and unconditional security in cryptography*, Ph. D. Thesis, ETH Zurich, Hartung-Gorre Verlag, Konstanz, 1997.