國立交通大學

資訊科學與工程研究所

碩士論文

敲鍵行為統計學習模型應用於網路身份認證

Keystroke Statistical Learning Model for Web Authentication

研 究 生：江檉皇

指導教授：謝續平 博士

中華民國九十五年六月

敲鍵行為統計學習模型應用於網路身份認證

Keystroke Statistical Learning Model for Web Authentication

研 究 生：	江樫皇	Student:	Cheng-Huang Jiang
指導教授：	謝續平 博士	Advisor:	Dr. Shiuh-Pyng Shieh

國立交通大學

資訊科學與工程研究所

碩士論文

A Thesis
Submitted to
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
In Partial Fulfillment of the Requirements
For the Degree of
Master
In

Computer Science and Information Engineering

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

敲鍵行為統計學習模型應用於網路身份認證

研究生：江樫皇　　　　　　　　　　指導教授：謝續平

國立交通大學　資訊科學與工程研究所

摘　要

傳統網路身份認證機制單純依靠檢查帳號和密碼的正確性，已經不足以應付急速發展的網路應用和快速成長的電子商務，如果發生使用者的帳號和密碼被他人竊取使用的情況，傳統網路身份認證機制將無法正確辨識出登入者的真實身份。敲鍵行為特徵分析屬於生物身份辨識科技的一種，具備低成本和透明性，相當適合用來搭配或取代傳統網路身份認證機制。本篇論文提出結合統計學習理論中的隱藏式馬可夫模型和高斯機率模型，來建立敲鍵行為特徵的統計機率模型，利用此統計機率模型來分析使用者登入帳號和密碼的敲鍵時間資訊，藉此提高登入者真實身份認證的準確性。實驗結果顯示，帳號和密碼的長度如果限制大於或等於九的話，本篇論文所提出的方法可以將錯誤率降低到 2.54 %。

Keystroke Statistical Learning Model for Web Authentication

Student: Cheng-Huang Jiang Advisor: Shiuh-Pyng Shieh

Department of Computer Science and Information Engineering
National Chiao Tung University

Abstract

As the rapid evolution of E-commerce, traditional password authentication mechanism is insufficient to provide strong security and reliability for identity verification of web-based applications. Under the circumstance that the intruder could make use of the username and password stolen from the innocent individual, conventional password authentication mechanisms are incapable to distinguish the discrepancy between the intruder and the innocent individual. Keystroke typing characteristics is one of the most novel and creative biometric techniques. The low-cost and transparency of keystroke typing characteristics make it appropriate to complement, but not to replace traditional password authentication mechanism used by web-based applications.

In this thesis, we proposed a statistical model for keystroke typing characteristics based on Hidden Markov Model and Gaussian Modeling from Statistical Learning Theory. The accuracy of the identity authentication can be substantially enhanced by analyzing keystroke timing information of the username and password using our proposed model. The results of the experiment showed that, with the condition on both the minimum length of the username and password restricted to be greater than or equal to 9, we achieved by far the best error rate of 2.54 %.

誌　謝

完成這篇論文，除了讓我學到研究的方法，也讓我得到很多的第一次，不管是好是壞，都讓我的人生得到了相當多的經驗。謝謝我的家人，在我陷入低潮的時候給我一個可以依靠的避風港，一直給我鼓勵和支持。謝謝ㄚㄚ在這段時間一直陪著我，讓我有動力完成論文。謝謝老師給我再一次的機會完成這篇論文。謝謝我的好朋友們的鼓勵和幫忙。謝謝網路上的匿名者們在實驗時提供的樣本。

謝謝大家，我要畢業了，喔耶！

Table of Contents

1. INTRODUCTION 1

1.1. Background 1

1.2. Contribution 3

1.3. Synopsis 3

2. RELATED WORK 5

2.1. Features 5

2.2. Performance Measures 5

2.3. Fixed-text Keystroke Analysis 7

2.4. Free-text Keystroke Analysis 11

2.5. Summary 13

3. MODELING AND METHODOLOGY 16

3.1. Features 16

3.2. Gaussian Modeling 17

3.3. Maximum Likelihood Estimation of the Parameters 18

3.4. Hidden Markov Model 18

3.5. Forward Algorithm 22

3.6. General Modules for Keystroke Analysis 23

3.7. Scheme and Measures 28

4. EXPERIMENTS AND RESULTS 31

4.1. Experiment Setting 31

4.2. Data Collection 31

4.3. Evaluation 32

5. CONCLUSIONS 41

6. REFERENCES 42

List of Figures

Figure 2.1: EER is the cross point at which the FRR and FAR are equal. 6

Figure 3.1: The Hidden Markov Model for keystroke analysis. 19

Figure 3.2: Graphical mode for digraph with keystroke sequence “banana” 20

Figure 3.3: Flow chart for Profile Building Module 25

Figure 3.4: Flow chart for Authentication Module 26

Figure 3.5: Flow chart for Identification Module 28

Figure 3.5: Flow chart of training phase for fixed-text keystroke analysis 29

Figure 3.6: Flow chart of recognition phase for fixed-text keystroke analysis 30

Figure 4.1: Target string length distribution of reference samples 32

Figure 4.2: O-Strategy - Minimum target string length = 9, reference sample size = 5 33

Figure 4.3: O-Strategy - Minimum target string length = 9, reference sample size = 10, EER = 5.71% 33

Figure 4.4: O-Strategy - Minimum target string length = 9, reference sample size = 15, EER = 5.24% 34

Figure 4.5: O-Strategy - Minimum target string length = 9, reference sample size = 20, EER = 4.76% 35

Figure 4.6: A-Strategy - Minimum target string length = 9, reference sample size = 5, EER = 6.19% 35

Figure 4.7: A-Strategy - Minimum target string length = 9, reference sample size = 10, EER =3.81% 36

Figure 4.8: A-Strategy - Minimum target string length = 9, reference sample size = 15, EER =2.91% 36

Figure 4.9: A-Strategy - Minimum target string length = 9, reference sample size = 20, EER =2.54% 37

Figure 4.10: O-Strategy – EER vs. Minimum target string length 38

Figure 4.11: A-Strategy – EER vs. Minimum target string length 38

Figure 4.12: O-Strategy: FRR with different number of reference samples 39

Figure 4.13: O-Strategy: FAR with different number of reference samples 39

Figure 4.14: A-Strategy: FRR with different number of reference samples 40

Figure 4.15: A-Strategy: FAR with different number of reference samples 40

1. Introduction

1.1. Background

As the web interface becoming more powerful and convenient, the trend has been shown that more and more applications are developed for web-based services instead of for local use only. Web-based services change the way people using computer, and makes it more easily for people to globally and ubiquitously acquire information and resources. Unfortunately, it also makes more chances for malicious attack and intrusion to be happened. As a consequence, guarantee the accuracy of the user identity for web-based services has became a significant issue.

Conventionally web-based services employ username/password pairs to authenticate the identities of the users. After the users pass the authentication phase, the systems assume the identities of the users are consistent and rarely use other mechanism to constantly assure the identities of the users. This comes up with two acute security issues. First one is at the authentication phase. An attacker can steal the username/password pair by any means from the user, say Alice, log on to the web-based services by claiming to be Alice, and gain access to the web-based service as Alice. Second one is after the authentication phase has been legally passed by Alice. An attacker can access the web-based services and act as Alice when she is temporarily leaving the computer without log out the systems or close the browser. Hence we need another security mechanism to complement or serve as a robust safeguard to prevent legal users from being impersonated and unauthorized.

Keystroke dynamics, also referred to as keyboard typing characteristics or keyboard typing rhythms, is one of the most novel and creative biometric techniques. There are two types of biometrics: physiological biometrics and behavioral metrics. Physiological biometrics requires user to provide a given physical characteristic in different positions and/or conditions, but always the same characteristic, such as fingerprint, facial recognition, hand geometry, iris scan, retinal scan, vascular patterns and DNA. Most of them require expensive hardware to support the dedicated function. As a result, they are impractical and inefficient to combine with the authentication mechanism of the web-based services. Behavioral biometrics requires user to behave in a consistent manner, including speaker recognition, keystroke dynamics, hand-writing and mouse movement. Keystroke dynamics has following advantages over others:

- It is non-intrusive, since user will be typing at the keyboard anyway.
- It is transparent, since keystroke patterns can be captured silently without interrupting user's normal activity.
- It is low-cost, since the hardware requirement is only the keyboard which is already presented, and the analysis can be conducted and implemented by software.

Keyboard dynamics has the disadvantage of instability inherited from behavioral biometrics while the people may behave differently and be influenced by the environment, physiology status or different keyboards. Other than that, keystroke dynamics is considered to be an economical and practical measure to be in conjunction with, or in place of traditional authentication method of web-based services.

Keystroke dynamics is based on the assumption that different people have unique habitual rhythm patterns in the way they type. Previous work ([19, 11]) has been shown that keystroke dynamics is good evidence of identity. Within the keystroke dynamics literatures, the research can be divided into two categories: fixed-text keystroke analysis and free-text keystroke analysis, according to the structures of the typing patterns to be analyzed. In the fixed-text keystroke analysis, the patterns are short, fixed and structured, such as login-password pairs at the authentication phase or the pass-phrase predetermined by the authentication system. The methods [2, 3, 4, 6, 9, 19] proposed for fixed-text keystroke analyses are usually encouraged to integrated with or replace of traditional web-based authentication method. As to the free-text keystroke analysis, the patterns are diverse and long. They can be collections of keystrokes from an email sending by an employee or anything a user typed in any period of time. Free-text keystroke analyses [1, 11, 15, 16] are suitable for continuously identity verification after the authentication phase has passed.

1.2. Contribution

In this thesis, we present a formal statistical model for keystroke dynamics analysis using Hidden Markov Model and Gaussian Modeling. Underlying the proposed model and the schemes for fixed-text keystroke analysis can be applied to web-based services as authentication mechanism for enhancing security strength according to different security requirements. The proposed model can be extended to devise scheme for free-text keystroke analysis. The experiment of our scheme for authentication resulted on 2.54 % of Equal Error Rate, which is the best so far in the literature.

1.3. Synopsis

This thesis is organized as follows. The related work of keystroke dynamics is presented in Chapter 2. In Chapter 3, formal model along with schemes for fixed-text keystroke analysis are proposed. Furthermore, the experiments and results are given in Chapter 4. Finally, a conclusion and future work is given in Chapter 5.

2. Related Work

In this chapter, we first review the features used to analyze by keystroke dynamics in Section 2.1, and the performance measurements for keystroke dynamics evaluation are presented in Section 2.2. Furthermore, we review the literature with regard to fixed-text keystroke analysis in Section 2.3. We present a review for free-text keystroke analysis in Section 2.4.

2.1. Features

There are several measurements can be used by keystroke dynamics analysis when the user press the keys on the keyboard. The possible measurements can be listed as follows:

- Keystroke duration: The time a key stays pressed.
- Keystroke latency: The time interval between two consecutive keystrokes (also referred to as digraph). It can be extended to N consecutive keystrokes (also referred to as *n*-graph).
- Keystroke frequency: the number of times the keystroke appeared.

First two of the list above is the most popular features used in the literature. The mean and standard deviation of the keystroke duration or latencies are the basis measurement to combine with other techniques for timing characteristics analysis.

2.2. Performance Measures

Typically biometric performance has three metrics for describing performance in terms of decision rates with regard to accuracy [21]:

- False Reject Rate (FRR): The expected portion of valid user attempts identified as imposters. A false rejection is also referred to as a “Type

I" Error in mathematical literature, or false alarm rate (FAR). The higher the FRR is, the lower the feasibility and convenience the systems will perform.

- False Accept Rate (FAR): The expected portion of imposter attempts identified as valid users. A false acceptance is also referred to as a "Type II" error in mathematical literature, or imposter pass rate (IPR). The higher the FAR is, the more opportunities the systems give the imposters to breach in.
- Equal Error Rate (EER): In Figure 2.1, the value of the cross point at which the FAR and FRR are equal for a determined threshold. The threshold is the parameters which can be adjusted for different security strength in the algorithms. EER is also referred to as crossover error rate (CER).
- Average False Rate (AFR): Average of FRR and FAR.

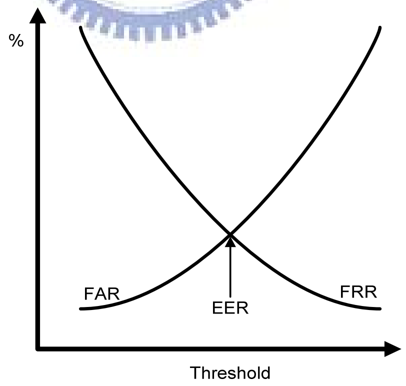


Figure 2.1: EER is the cross point at which the FRR and FAR are equal.

In most of the literature, the terms false reject rate and false accept rate are used interchangeably with false alarm rate and imposter pass rate. For the sake of clarity, in this thesis we will use FRR to denote the false reject rate, FAR to denote the false accept rate, and ERR to denote the equal error rate respectively.

2.3. Fixed-text Keystroke Analysis

In the keystroke dynamics literature with regard to fixed-text keystroke analysis, the typing patterns to be analyzed are short, fixed, predetermined, and structured. The research can be separated into two portions based on the concerns the approaches presented. One portion puts their focuses on designing approaches which can be in conjunction with, or in place of traditional authentication mechanism by analyzing the keystroke timing information of username/password pairs typed by individuals. The other portion tries to figure out how to identify the user's identity from keystroke timing information of predetermined texts typed by individuals for certain times. In the Section 2.3.1, we introduce the algorithms for keystroke timing information analysis of login-password. In the Section 2.3.2, the review of the keystroke analysis methods for predetermined text is presented.

2.3.1. Traditional login-password Authentication Mechanism

Joyce and Gupta [19] proposed a simple and promising approach to analyze four target strings (login, password, first name, last name) during a login process. Their system requires new users to type the reference signatures in term of four target strings for eight times. The mean reference signature is then given by:

$$M = \{M_{username}, M_{password}, M_{firstname}, M_{lastname}\}$$

At the authentication phase, a test signature T is presented and compares with reference signature M to determine the magnitude of the difference between M and T. Let $M = (m_1, m_2, \ldots, m_n)$ and $T = (t_1, t_2, \ldots, t_n)$ where n is the total number of latencies in the signature, the verifier computes the magnitude of the difference between M and T as the l_1 norm:

$$\left\| M - T \right\|_1 = \sum_{i=1}^{i=n} \left| m_i - t_i \right|$$

Then a suitable threshold for an acceptable size of the magnitude is chosen for each user based on a measure of the variability of user's signatures. The mean and standard deviation of the norms $\left\| M - S_i \right\|$, where S_i is one of the training signatures, are used to decide a threshold for an acceptable norm value of the latency difference vector between a given M and T. If this norm is less than the threshold for the user, the attempt is accepted. Otherwise it is flagged as an imposter attempt. Thirty-three users were participated in the evaluation. 13.3% (4 out of 30) FRR and 0.17% (1 out of 600) FAR were obtained. EER is not available because they did not conduct the experiment for every possible threshold values.

Magalhaes et al [6, 3] proposed a lightweight algorithm to analyze only one target string with password or pass-phrase. Each user has to type password or pass-phrase for twelve times to form the reference profile. They enhanced [6] based on [3] by integrated the concept of keyboard gridding in [5]. By using the concept of keyboard gridding, their algorithm is specifically designed and optimized for right-handed users. As a consequence, the algorithm they proposed can not ensure the same results on left-handed users. 5.58 % EER was obtained in [6], and less than 5% EER was achieved in [3].

Ru and Eloff [4] used fuzzy logic to characterize the typing behavior of the users based on the keystroke latencies, the distance of the keys on the keyboard, and typing difficulty of the key combinations. Twenty-five samples are required for enrollment. Username and password are used as target strings to be analyzed. 7.4% FRR and 2.8% FAR were obtained in the experiment [2]. EER is not available because they did not conduct the experiment for every possible threshold values.

Haidar et al [9] presented a suite of techniques using neural networks, fuzzy logic, statistical methods, and several hybrid combinations of these approaches to learn the typing behavior of a user. In the experiment, 2% FRR and 6% FAR were obtained [2].

Bleha et al [23] proposed two approaches for authentication using minimum distance classifier and Bayesian classifier. The normalized minimum distance classifier was

$$D_i = \frac{(X - m_i)^t (X - m_i)}{\|X\| \cdot \|m_i\|}$$

, and the normalize Bayesian classifier was

$$d_i = \frac{(X - m_i)^t C_i^{-1} (X - m_i)}{\|X\| \cdot \|m_i\|}$$

, where the participant is claiming to be user i, X is the latencies vector, m_i is the latency means of the reference samples and C_i is the latencies covariance of the reference samples. Both classifiers have defined thresholds for deciding the acceptance of the user. They din not mention the results would come up with while different threshold values were used. In the experiment, 8.1% FRR and 2.8% FAR were obtained. EER is not available because they did not conduct the experiment for every possible threshold values.

2.3.2. Predetermined Text

Gaines et al [18] conducted an experiment in which seven professional secretaries were asked to type three passages, about 300-400 words long (the first one is ordinary English text, the second one is the collection of random words, and the third one is the collection of random phrases), for two different times separately within four months. Keystroke latency of the digraph that appeared more than ten times were computed for each individual. A classical two-sample t-test of statistical independence on the hypothesis that the percentage of the means and variances of the digraphs appeared in both sessions were the same that passed the test were between 80 and 95. Outliers were removed and the rest of data were transformed by logarithm. An assumption was made that the raw data was log-normally distributed, and the transformed data was observed to be approximately normally distributed. 4% FRR (2 out of 55) and 0% FAR were obtained. However, the number of volunteers was too small and the amount of data required to create the reference profiles was insufficient. Only twenty-six lower-case letters and space key were taken into consideration, resulted that only 27 * 27 = 720 different digraphs. Additionally, since the limitation by the length and content of the three passages, only 87 combinations of digraph were analyzed in the experiments. Consequently, the results of FAR and FRR obtained in the experiment resulted on a lower confidence.

Leggett and Williams [8] proposed an improved approach based on [20]. They reported the results of two experiments similar to the experiment conducted in [18]. In the first experiment, seventeen programmers, with different typing ability, each one provided two samples. First sample was 1400 characters long served as the reference profile, and second one was 300 characters long served as the test profile. In the second experiment, thirty-six participants provided two passages in 537 characters long in two months separately with a delay of at least one month. Their approach compares all digraph latencies between all combinations of digraphs in the samples. The test digraph was classified as valid one if the test digraph latency was within 0.5 standard deviation of the reference digraph latency mean. 5.5% FRR and 5% FAR were obtained.

Bergadano et al [17] proposed an approach which measure digraph latencies based on the degree of disorder. Given two typing samples of the same text, the digraphs shared between both typing samples are retrieved, and the durations of *n*-graphs are computed. The mean of duration is calculated if *n*-graph is reported more than one time. Then the shared *n*-graphs in both typing samples are sorted by the duration and stored in two arrays respectively. The degree of disorder is computed as the sum of the distances between the positions of each *n*-graph in both sorted array. The predetermined sample texts in the experiment are a passage of one famous Italian novel plus a short text in English. Each sample was produced using only twenty-six lower-case letters, plus the space, the full stop, the comma, the apostrophe and the carriage return keys. 154 volunteers were involved in the experiments. 4% FRR and less than 0.01% FAR were achieved.

2.4. Free-text Keystroke Analysis

Monrose and Rubin [11] proposed the method using Euclidean Distance and probabilistic calculations on structured and unstructured texts. They made the assumption that the digraph latencies exhibit Gaussian distribution similar to the assumption in this thesis. As to the dynamic identity verification, the author pointed out that their approaches were not qualified to authenticate the user during the lifetime of a login session. Their approaches would fall hopelessly behind as the new measurements arriving almost each second, since the continuous nature of the problem and the expensive computation of their approaches. The experiments required forty-two users to type a few sentences from a list of available phrases and a few unstructured sentences. 90% of accurate classification rate was achieved on structured texts, and only 23% of accurate classification rate was obtained on unstructured texts.

Dowland et al [15, 16] reported a preliminary result on determining which approaches provide the basis for further research on continuous authentication with keystroke timing characteristics. Statistical analysis and data-mining analysis were used to investigate. 50% of accurate classification rate was obtained in [15], and 60% of accurate classification rate was improved in [16].

Gunetti and Picardi [1] improved the approach proposed in [17] and introduced two measures, R measure and A measure, to compute the degree of disorder between two samples of free texts. R measure was the one described in [17]. The authors concluded that the length of two typing samples must be long enough to gather enough shared *n*-graphs between two typing samples, resulted on the typing patterns to be analyzed were free text. If the number of shared *n*-graph was not sufficient, the degree of disorder computed for the free texts is less representative. A measure is to suppose that the durations of the same *n*-graph from two sample texts which came from the same individual are similar. The author combined two proposed measure with different *n*-graph to conduct the experiments. In the experiments, forty volunteers, each one provided 15 samples for building reference profiles. Another one hundred and sixty-five people, each one provided one samples as imposters. All of them were native Italians and the samples were written in Italian. No more than one sample provided each day via browser, and the samples were collected on the basis of the availability and willingness of volunteers over a period of six month. Each size of the samples is about 700 to 900 characters long. The limitation of their approach is that the length of the samples must be long enough. Furthermore, their approach is computationally expensive, and it is infeasible to be applied on verification of identity via keystroke analysis in real time.

2.5. Summary

After the review of the keystroke dynamics analysis in the literature so far, we can see that the most common method the different approaches used as basis is that most of them leveraged the mean and standard deviation of the digraph latencies or durations. The approaches determined the valid attempts by checking whether the latencies or durations of the digraphs typed by the volunteers fall within the standard variation of the digraph latencies or duration mean as follows:

$$D_\mu^p - wD_\sigma^p \le D \le D_\mu^p + wD_\sigma^p$$

, where D is one of the digraph latencies or durations in the test profile, D_μ^p and D_σ^p are the corresponding digraph in the individual's reference profile, and w is the weighting factor.

There are several issues concerned by keystroke analysis for it to be feasible and practical as follows.

- The length of target strings to be analyzed. The longer the target string is, the more the digraphs can be used to compare between reference profiles and test profiles. By the nature limitation of traditional authentication method with username/password, it is difficult to force the users to remember their account information in terms of username and password with ten characters long respectively or even more. This limitation lead to the situation that the methods proposed in the literature can only have spare timing information to analyze.
- The number of samples to construct the reference profile. The purpose

of calculating the mean and standard deviation of the digraph latencies and duration is to model the personal typing characteristics in statistics. The more the number of sampled provided, the higher the accuracy of the mean and variance is close to the actual behavior of individual's typing characteristics. However, the number of samples also influences the feasibility of the methods in practical. In static keystroke analysis, all methods required the volunteers to type the same structured or predetermined texts for several times. The number of repetition times is inverse propositional to the desire of the volunteers and the users to adopt the proposed methods.

Previous works, regarding continuously monitor and analyze the typing patterns for real time identification of individuals have not yielded satisfied results. Even if the accuracy is acceptable, it takes too long to recognize the identities of individuals. Consequently, it is infeasible in the environment with high security level, such as government and military computers, since the confidential information could have been stolen for long enough without raising the alarm. As the result, the time it takes for identification is a critical issue. In the literature with regard to this portion, the computation costs of the proposed algorithms are high. As a result, it is a problem to verify the user's identity in real time. Unlike the limitation in static keystroke analysis, the sample texts can be of any length, usually longer enough to have sufficient number of digraph combinations and repetition times for calculating more accurate mean and standard deviation to fit the individual's typing behavior. But it has corresponding limitation that the reference samples and the test samples could be two totally different texts. The situation could happen while the digraphs in the test samples are not existent in the reference samples. The literatures regarding free-text keystroke analysis have not discussed this issue, either they made the assumption that both the length of reference samples and test samples must be longer enough to exist shared digraph, or made the constraint that only twenty-six lower-case letter plus several punctuation marks were taken into consideration to construct the text to analyze.

As to the experiment setting, it is very difficult to obtain a sample representative of the population, since we do not know how to characterize the population [6]. The experiment setting in each of the proposed method is different, such as the number of the volunteers involved, the length of the target strings, the number of samples to construct the reference profiles, and where the experiments were conducted. With all these variations resulted on that there is no fair standpoint to compare the results achieved by each method.

3. Modeling and Methodology

In this chapter, we first make a formal definition of features we will use for keystroke analysis in Section 3.1. The Gaussian Modeling for *n*-graph timing information is presented in Section 3.2, and we introduce how to calculate parameters for Gaussian Modeling with Maximum Likelihood Estimation in Section 3.3. In Section 3.4, the Hidden Markov Model is used to model the timing information given a keystroke sequence, and the Forward algorithm used to calculate the probability given a keystroke sequence is presented in Section 3.5. In Section 3.6, we devise three general modules for keystroke analysis. Furthermore, we introduce the schemes for keystroke analysis in Section 3.7.

3.1. Features

In the related work, we have reviewed the features used to analyze keystroke dynamics. In this section, we will give more formal definition on the features that will be taken into consideration in our model. A single keystroke will trigger two events, the key press event and the key release event, along with the time while both events occurred. Denote the set of keys of interest as Q, and let $|Q|$ denote the cardinality of the set Q.We define the time point while the key is pressed as $TKS_{pressed}$. We refer to N consecutive keystrokes as *n*-graph. The special case of single keystroke is referred to as unigraph. Two consecutive keystrokes are referred to as digraph in the literatures, and trigraph means three consecutive keystrokes, etc. Given a sequence of consecutive keystrokes $S = \{s_1, s_2, \ldots, s_m\}$, where m is the number of keystroke sequence, we have *n*-graph with the size of $m - n + 1$. We define the duration of *n*-graph

$GD=\{d_1,d_2,d_3,\ldots,d_k\}_{k\in N,1\le k\le m-n+1}$ as follows:

$$d_k = TKS^{s_{n+k-1}}_{pressed} - TKS^{s_k}_{pressed}\ .$$

The durations of *n*-graph are used as timing features for further analysis in our model.

3.2. Gaussian Modeling

Previous work [10, 11, 13, 18] have shown that the durations distribution of a given set of digraphs forms an approximate Gaussian distribution. Therefore we make a natural assumption that the *n*-graph $q \in Q^n$, with duration y, $\Pr[y \mid q]$, forms a Gaussian distribution, such that

$$\Pr[y \mid q] = \frac{1}{\sqrt{2\pi}\sigma_q} e^{-\frac{(y-\mu_q)^2}{2\sigma_q{}^2}} \quad ,$$

where μ_q is the mean value of the duration y for *n*-graph q, and σ_q is the standard deviation. Since behavioral characteristics of the individuals could be influenced by many reasons, the statistical analysis method used by previous work can be viewed as the same probability was given to the valid attempts of digraph latencies and durations within the standard deviations of the mean durations. By using Gaussian Modeling, we can give higher probability to the *n*-graph durations of test samples that is more close to the *n*-graph mean durations of reference samples, and lower probability to the *n*-graph duration that is far from the mean of the *n*-graph for the reason that the individuals could be temporarily out of regular typing behavior, and we can take the irregular typing behavior without discarding the possibility that the set of *n*-graph durations provided by the corresponding individuals.

3.3. Maximum Likelihood Estimation of the Parameters

With the limitation that we are unable to collect all the typing keystrokes of the individual and calculate the exact parameters of the means and variances for each distinct combination of *n*-graph durations. We have to deduce $\{\ (\hat{\mu}_q, \hat{\sigma}_q)\ \}_{q \in Q^n}$ of *n*-graph durations, give a keystroke sequence S, by the method of maximum likelihood estimation of the parameters. Fortunately, the maximum likelihood estimation of the parameters for Gaussian distribution can compute the sample mean and sample variance as follows.

$$\hat{\mu}_q = \frac{\sum_{i=1}^{k} d_i(q)}{k}$$

$$\hat{\sigma}_q^2 = \frac{\sum_{i=1}^{k} \left[d_i(q) - \hat{\mu}_q \right]^2}{k-1}$$

, where k is the number of *n*-graph q appeared in S.

3.4. Hidden Markov Model

Hidden Markov Models (HMMs) [12, 14, 22] are proper for modeling sequential data, such as the sequence of keystroke timing information that we take into consideration in this thesis. HMMs have been widely applied in areas such as speech recognition, optical character recognition, machine translation, bioinformatics, and genomics. A Markov process is a stochastic process with the property that the probability of transitioning from previous state to current state depends only on the previous state and was independent of all other previous states. In general, a Markov model is a way of describing a process that goes through a series of states [14]. In a general Markov Model, the state is directly observed by the observer. In a Hidden Markov Model, the state is not directly visible, and some outputs from the state are observed. A Hidden Markov Model can be viewed as a chain of mixtures models with unknown parameters.

The HMM we use to model the timing information of keystroke sequence is shown in Figure 3.1. It is a statistical graphical model, where each circle is a random variable. Unshaded circles q_t represent are unknown (hidden) state variables we wish to infer, and shaded circles y_t are observed state variables, where t is a specific point in time. A is a state transition matrix holding the probabilities of transitioning from q_t^i to q_{t+1}^j, where q^i (or q^j) means the *i*-th (or *j*-th) state. So we have $P\left(q_{t+1}^j = 1 \mid q_t^i = 1\right) = A_{ij}$. η is an state emission matrix holding the output probability $P\left(y_t \mid q_t^i = 1\right)$ of *i*-th state. π_i is the initial state probability of *i*-th state. A compact notation $\lambda = (A, \eta, \pi)$ is used to indicate the complete parameter set of the model.

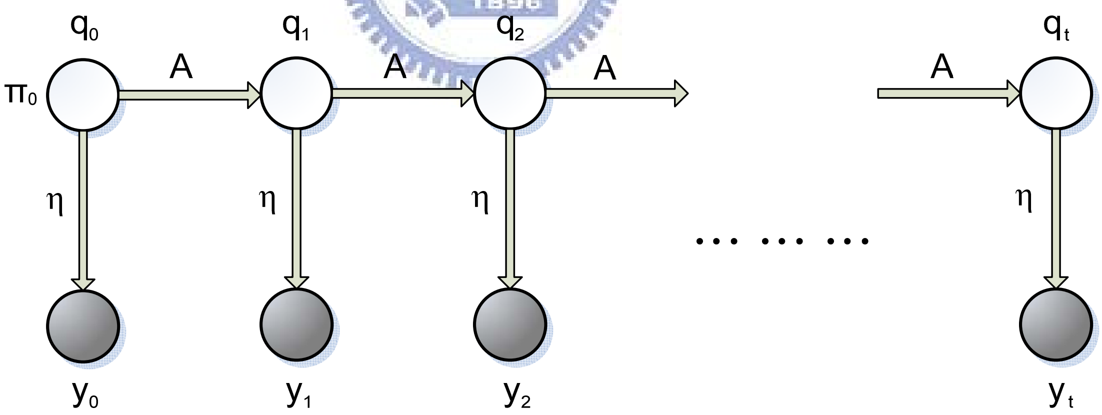


Figure 3.1: The Hidden Markov Model for keystroke analysis.

In our setting, given a keystroke sequence S, *n*-graph G, [*n*+*1*]-graph G', such that

$$
\begin{aligned}
S &= \{s_1, s_2, \dots, s_m\}_{m \in N} \\
G &= \{g_1, g_2, g_3, \dots, g_{m-n+1}\} \, . \\
G' &= \{g'_1, g'_2, g'_3, \dots, g'_{m-n}\}
\end{aligned}
$$

The state transition matrix A is the probability of the frequency that the [$n+1$]-graph appeared in the S as follows.

$$A_{g_t, g_{t+1}} = \frac{|g'_t|}{m-n}$$

For instance in Figure 3.2, given a keystroke sequence “banana” and digraph is of interest, the digraph “na” is following the digraph “an”. We have 5 (6-2+1) digraphs in “banana”, 4 (6-3+1) trigraph in “banana”, and the trigraph “ana” appears two times. As a result, we have the transition probability of $2/4 = 0.5$ from “an” to “na”.

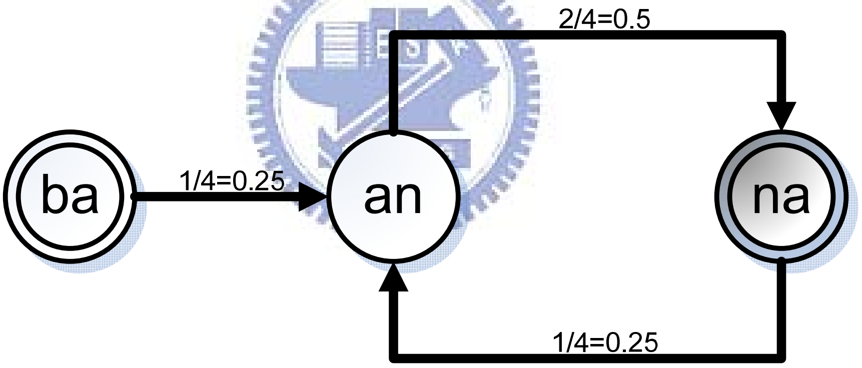


Figure 3.2: Graphical mode for digraph with keystroke sequence “banana”

The state emission matrix η here is defined as the Gaussian distribution probability of the n-graph $G = \{g_1, g_2, g_3, \dots, g_{m-n+1}\}$ with duration $GD = \{d_1(g_1), d_2(g_2), d_3(g_3), \dots, d_{m-n+1}(g_{m-n+1})\}$ as follows.

$$\eta_g(d(g'))=\begin{cases}\Pr[\,d(g')\,|\,g\,]=\dfrac{1}{\sqrt{2\pi}\,\sigma_g}e^{-\frac{[d(g')-\mu_g]^2}{2\sigma_g{}^2}} & ,g=g' \\ 0,g\neq g' & \end{cases}$$

For example, given a sample duration 80ms of digraph “na” with the mean 100ms and the standard deviation 30ms, we can calculate the emission probability of sample digraph duration as follows.

$$\eta_{"na"}(80ms)=\Pr[100ms\,|\,"na"]=\frac{1}{\sqrt{2\pi}\cdot 30}e^{-\frac{(80-100)^2}{2\cdot 30^2}}=0.010648267$$

The initial probability vector π is the probability of the frequency that the *n*-graph appeared in the S.

There are three basic problems to solve with the HMM $\lambda=(A,\eta,\pi)$. These problems are the following.

- Given a model parameters $\lambda=(A,\eta,\pi)$ and observation output sequence $O=O_1O_2O_3\ldots O_t$, compute the probability $P(O\,|\,\lambda)$ of the observation output sequence.
- Given a model parameters $\lambda=(A,\eta,\pi)$ and observation output sequence $O=O_1O_2O_3\ldots O_t$, find the most probable state sequence $Q=Q_1Q_2Q_3\ldots Q_t$ which could have generated the observation output sequence.
- Given a observation output sequence $O=O_1O_2O_3\ldots O_t$, generate a HMM $\lambda=(A,\eta,\pi)$ to maximize the $P(O\,|\,\lambda)$.

We make the assumption that each individual has his/her own HMM with $\lambda=(A,\eta,\pi)$ for individual’s keystroke timing characteristics. The problem to solve is that, given a keystroke sequence S and its timing information, we have to choose one from the number of HMMs which has the highest probability to generate the keystroke sequence S. Consequently, first we have to calculate the probability of keystroke sequence S for each HMM. This is similar to the first basic problem to solve with HMM as described above, and we will show how to solve the problem with Forward algorithm in the next section.

3.5. Forward Algorithm

The problem of finding the probability of keystroke sequence S can be viewed as how well a given HMM $\lambda = (A, \eta, \pi)$ would score on S. We use the Forward algorithm [22] to calculate the probability of a m long keystroke sequence S with *n*-graph G, and *n*-graph durations GD,

$$\begin{aligned} S &= \{s_1, s_2, \ldots, s_m\}_{m \in N} \\ G &= \{g_1, g_2, g_3, \cdots, g_{m-n+1}\} \\ GD &= \{d_1, d_2, d_3, \cdots, d_{m-n+1}\} \end{aligned}$$

The state probabilities $\alpha' s$ of each state can be computed by first calculating α for all states at $t = 1$.

$$\alpha_1(g_1) = \pi(g_1) \cdot \eta_{g_1}(d_1)$$

Then for each time step $t = 2, \ldots, k$, the state probability α is calculated recursively for each state.

$$\alpha_{t+1}(g_{t+1}) = \alpha_t(g_t) \cdot A_{g_t, g_{t+1}} \cdot \eta_{g_{t+1}}(d_{t+1})$$

Finally, the probability of keystroke sequence S given a HMM $\lambda = (A, \eta, \pi)$ is as follows.

$$\Pr[S, G, GD \mid \lambda] = \alpha_k(g_k) = \alpha_{k-1}(g_{k-1}) \cdot A_{g_{k-1}, g_k} \cdot \eta_{g_k}(d_k)$$

The Forward algorithm described above has certain difference from the original one in [22]. The emission probabilities take less computation to obtain since we use the Gaussian distribution to model observed states. Additionally the observed states are only connected to the corresponding unknown states because we know the exact combination of *n*-graph the individual typed. So the summation of all partial probability of the state at time t is ignored and only one probability is calculated.

In original version of the Forward algorithm, the computation involved in the calculation of $\alpha_t(j)$, $1 \le t \le T$, $1 \le j \le N$, where T is the number of observations in the sequence and N is the number of states in the model, requires $O(N^2T)$ calculations. In our modified version of the Forward algorithm, we can see that it only requires $O(NT)$ calculations.

3.6. General Modules for Keystroke Analysis

In general, there are two problems can be solved using our model.

- Given a keystroke sequence S, and a HMM λ describing individual's keystroke timing information, we wish to determine whether S come from λ or not. (Authentication)
- Given a keystroke sequence S and a set of HMMs $\lambda's$ describing different individuals' keystroke timing information, we wish to know which HMM most probably generated S. (Identification)

The first problem is that, given a test sample of keystroke sequence and a reference profile, we have to decide whether the sample belongs to the reference profile or not. The second problem is very similar to the solutions provided by physiological biometrics. In this section, we devise three modules: Profile Building Module, Authentication Module and Identification Module underlying the model and algorithm described in the previous sections.

In the Profile Building Module, first we have to build the reference profile for each user. It requires the user to provide the reference samples. The more quantity of reference samples provided, the more exact parameters can be extracted. After collecting sufficient number of reference samples, we use the maximum likelihood estimation for Gaussian Modeling to calculate the parameters of each *n*-graph duration. We also have to compute the transition probability matrix and initial probability vector with respect to Hidden Markov Model. Then the parameters calculated for Hidden Markov Model are treated as the base element of the reference profile for each user. The flow chart of Profile Building Module is shown in Figure 3.3.

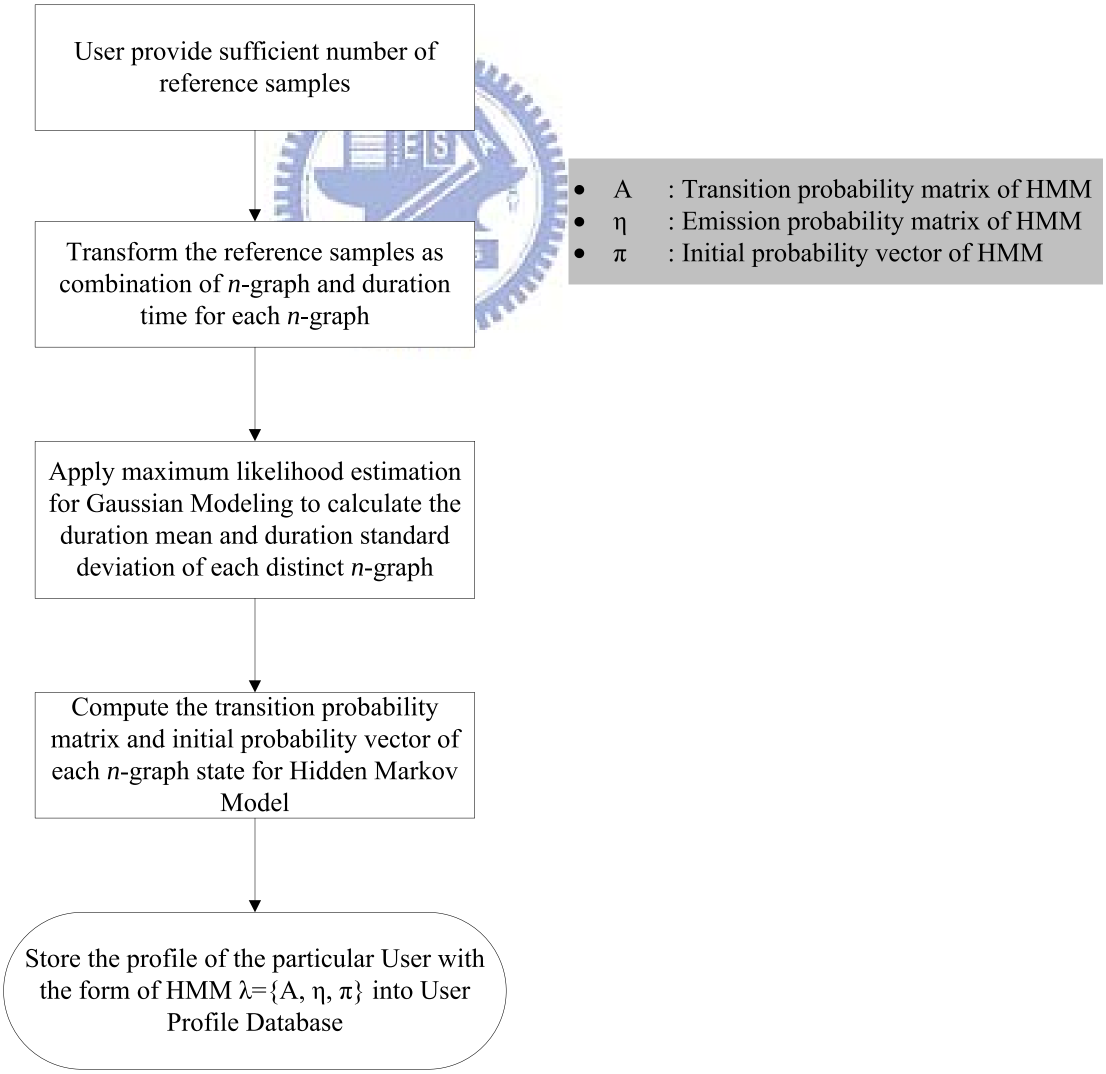

User provide sufficient number of reference samples
Transform the reference samples as combination of *n*-graph and duration time for each *n*-graph
Apply maximum likelihood estimation for Gaussian Modeling to calculate the duration mean and duration standard deviation of each distinct *n*-graph
Compute the transition probability matrix and initial probability vector of each *n*-graph state for Hidden Markov Model
Store the profile of the particular User with the form of HMM λ={A, η, π} into User Profile Database
• A : Transition probability matrix of HMM
• η : Emission probability matrix of HMM
• π : Initial probability vector of HMM

Figure 3.3: Flow chart for Profile Building Module

In the Authentication Module, given a keystroke sequence S of target string from a user with claimed identity ID, we wish to examine the possibility that S generated by ID. First we transform the keystroke sequence S to *n*-graph combinations G and calculate the timing information of *n*-graph duration GD as usual. At this moment, we have $S=\{s_1,s_2,\ldots,s_m\}_{m\in N}$, $G=\{g_1,g_2,g_3,\ldots,g_{m-n+1}\}$ and $GD=\{d_1,d_2,d_3,\ldots,d_{m-n+1}\}$. Now we produce a vector GDT, such that

$$GDT=\{\mu_{g_1}-\varepsilon\sigma_{g_1},\mu_{g_2}-\varepsilon\sigma_{g_2},\ldots,\mu_{g_{m-n+1}}-\varepsilon\sigma_{g_{m-n+1}}\}$$

, where ε is the weighting factor, μ_{g_k} is ID's duration mean of *n*-graph g_k, and σ_{g_k} is ID's duration standard deviation of *n*-graph g_k. GDT is the *n*-graph duration vector to evaluate the threshold value of the probability produced by modified Forward algorithm. With the inputs GD, GDT, and λ_{ID}, we can apply modified version of Forward algorithm to obtain two probability value $\Pr[S,G,GD\,|\,\lambda_{ID}]$ and $\Pr[S,G,GDT\,|\,\lambda_{ID}]$. $\Pr[S,G,GDT\,|\,\lambda_{ID}]$ can be viewed as the possibility if all the *n*-graphs durations in G are deviating ε times of duration σ from duration μ. $\Pr[S,G,GDT\,|\,\lambda_{ID}]$ is the threshold value of probability used to decide that the acceptance of the keystroke sequence S is confirmed if following expression is true.

$$\Pr[S,G,GD\,|\,\lambda_{ID}]\geq\Pr[S,G,GDT\,|\,\lambda_{ID}]$$

The weighting factor ε can be specified with respect to different level of security strength. The flow chart of Authentication Module is shown in Figure

3.4.

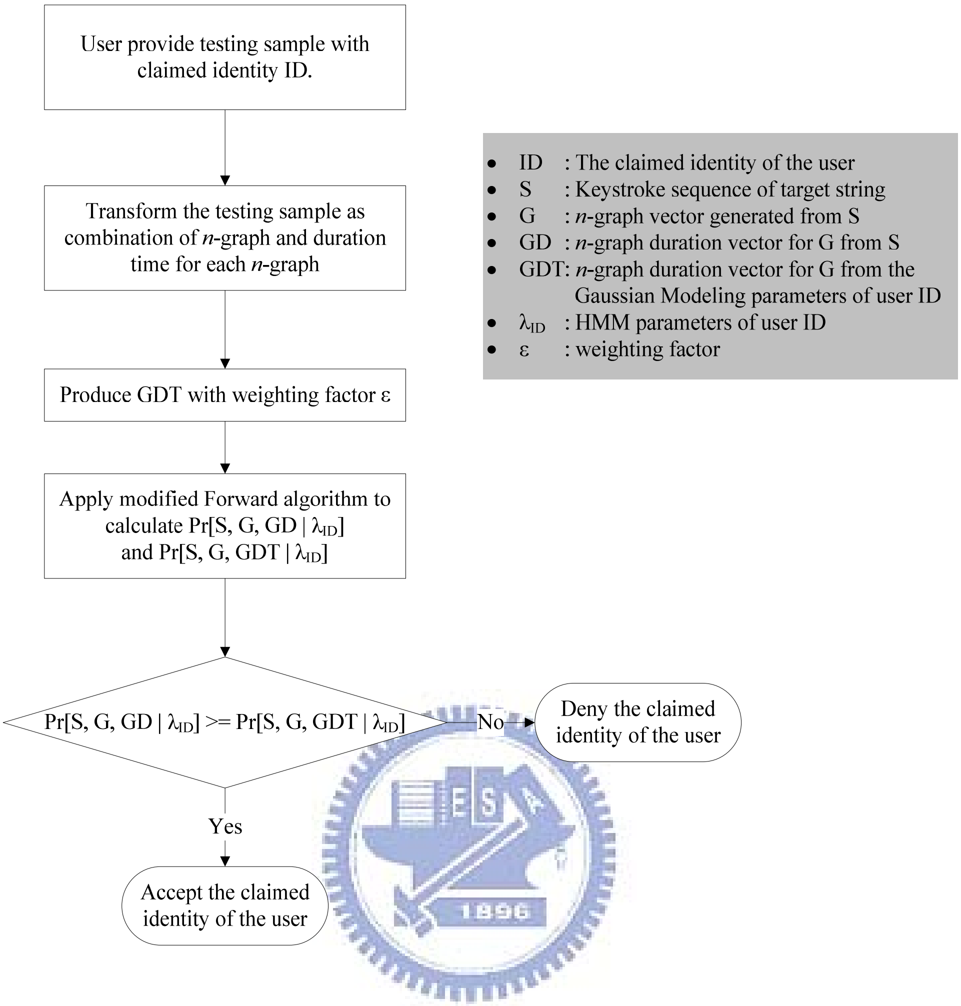

User provide testing sample with claimed identity ID.
Transform the testing sample as combination of n-graph and duration time for each n-graph
Produce GDT with weighting factor ε
Apply modified Forward algorithm to calculate Pr[S, G, GD | λID] and Pr[S, G, GDT | λID]
Pr[S, G, GD | λID] >= Pr[S, G, GDT | λID]
No
Deny the claimed identity of the user
Yes
Accept the claimed identity of the user
ID : The claimed identity of the user
S : Keystroke sequence of target string
G : n-graph vector generated from S
GD : n-graph duration vector for G from S
GDT: n-graph duration vector for G from the Gaussian Modeling parameters of user ID
λID : HMM parameters of user ID
ε : weighting factor

Figure 3.4: Flow chart for Authentication Module

In the Identification Module, given a keystroke sequence $S=\{s_1,s_2,\ldots,s_m\}_{m\in N}$ from the individual and a set of HMMs $\lambda's=\{\lambda_1,\lambda_2,\lambda_3\ldots,\lambda_l\}$, where l is the number of HMM. The problem is to choose the best one from $\lambda's$ which most probably generated S or there is no such one existed. In the beginning, the keystroke sequence S is transformed to n-graph combinations $G=\{g_1,g_2,g_3,\ldots,g_{m-n+1}\}$ and the timing information of n-graph duration $GD=\{d_1,d_2,d_3,\ldots,d_{m-n+1}\}$ is calculated. $\Pr[S,G,GD\mid\lambda_j]_{j\in N,1\le j\le l}$ for each HMM in $\lambda's$ is produced by modified

Forward algorithm. We select user U with the maximum probability over others', such as

$$\Pr[S, G, GD \mid \lambda_U] = \max\left(\Pr[S, G, GD \mid \lambda_j]_{j \in N, 1 \le j \le l}\right)$$

After that, we produce a vector GDT for user U, such that

$$GDT^U = \{\mu_{g_1}^U - \varepsilon\sigma_{g_1}^U, \mu_{g_2}^U - \varepsilon\sigma_{g_2}^U, \ldots, \mu_{g_{m-n+1}}^U - \varepsilon\sigma_{g_{m-n+1}}^U\}$$

, where ε is the weighting factor, $\mu_{g_k}^U$ is U's duration mean of *n*-graph g_k, and $\sigma_{g_k}^U$ is U's duration standard deviation of *n*-graph g_k. Again we use the modified Forward algorithm to calculate the $\Pr[S, G, GDT^U \mid \lambda_U]$. If the expression $\Pr[S, G, GD \mid \lambda_U] \ge \Pr[S, G, GDT^U \mid \lambda_U]$, the keystroke sequence S generated by user U is confirmed. Otherwise, we consider the keystroke sequence S is not generated by any user in the User Profile Database. The flow chart is shown in Figure 3.5.

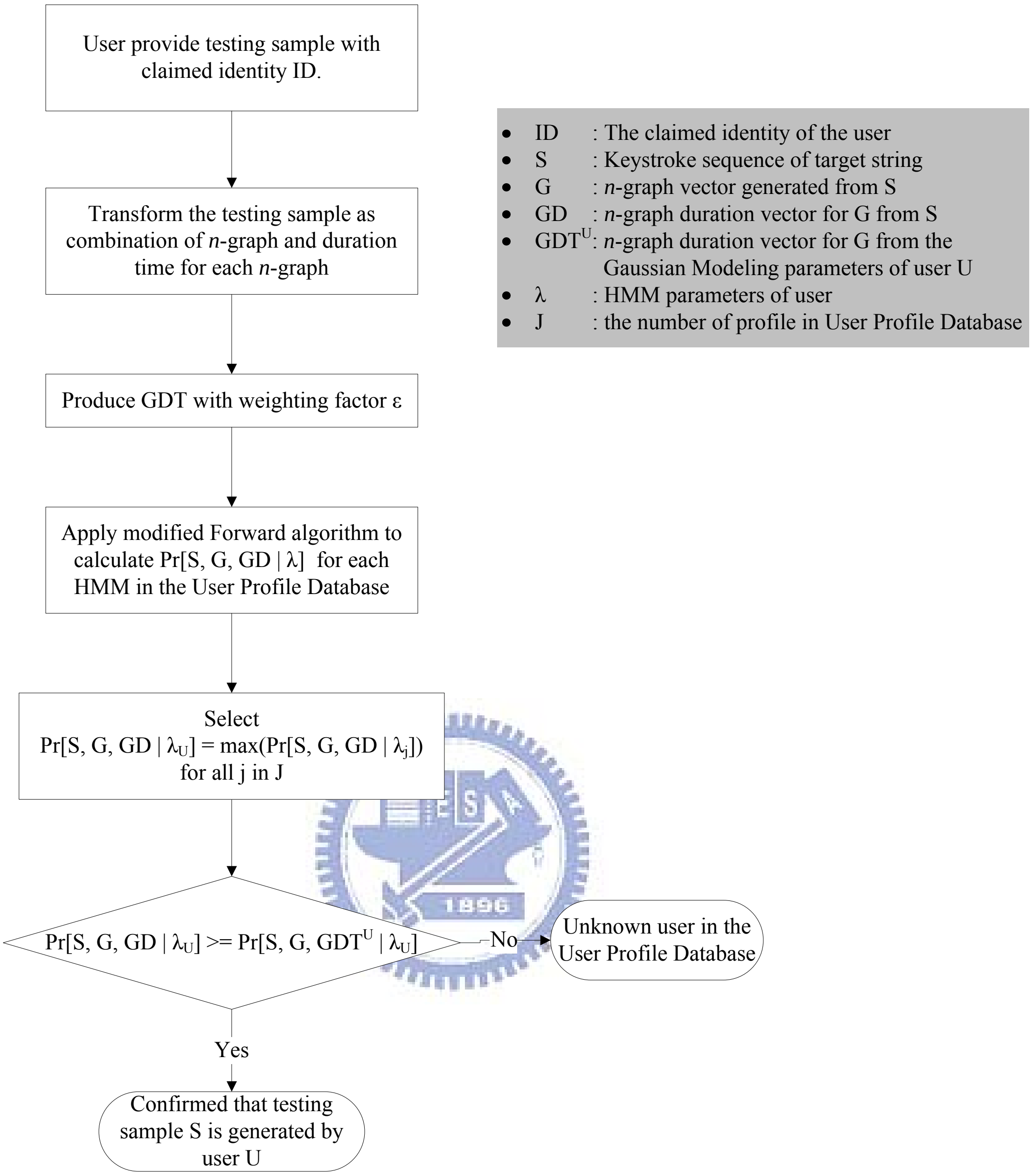

User provide testing sample with claimed identity ID.
Transform the testing sample as combination of n-graph and duration time for each n-graph
Produce GDT with weighting factor ε
Apply modified Forward algorithm to calculate Pr[S, G, GD | λ] for each HMM in the User Profile Database
Select Pr[S, G, GD | λU] = max(Pr[S, G, GD | λj]) for all j in J
Pr[S, G, GD | λU] >= Pr[S, G, GDTU | λU]
No
Unknown user in the User Profile Database
Yes
Confirmed that testing sample S is generated by user U
ID : The claimed identity of the user
S : Keystroke sequence of target string
G : n-graph vector generated from S
GD : n-graph duration vector for G from S
GDTU: n-graph duration vector for G from the Gaussian Modeling parameters of user U
λ : HMM parameters of user
J : the number of profile in User Profile Database

Figure 3.5: Flow chart for Identification Module

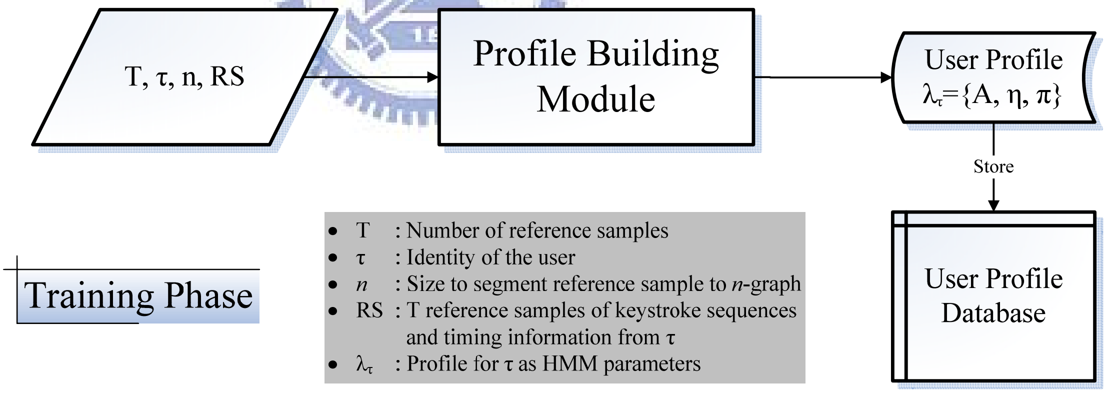
3.7. Scheme and Measures

Within the literature of fixed-text keystroke analysis, most of the proposed approaches put the emphasis on the application of authentication. There are several aspects to be concerned as follows:

- The target string to be analyzed could be username, password, first name, last name, or pass-phrase, which are normally short and usually three to sixteen characters in length.
- The samples used to build the reference profiles and the samples used to compare are identical and fixed strings, the difference is the timing information extracted from them.

We devise the scheme for fixed-text keystroke analysis according to the concerns listed above. There are two phases in the scheme for static keystroke analysis, the training phase and the recognition phase. The training phase is to build the user profiles as the database for recognition phase to compare with.

In the training phase, we have to decide the number of reference samples from each target string, and the size of *n*-graph to segment the target string. Figure 3.5 depicts the process of training phase.



T, τ, n, RS
Profile Building Module
User Profile λ_τ={A, η, π}
Store
User Profile Database
Training Phase
• T : Number of reference samples
• τ : Identity of the user
• *n* : Size to segment reference sample to *n*-graph
• RS : T reference samples of keystroke sequences and timing information from τ
• λ_τ : Profile for τ as HMM parameters

Figure 3.5: Flow chart of training phase for fixed-text keystroke analysis

In the recognition phase, we divide it into two parts according the function for dedicated requirement: Authentication or Identification. Figure 3.6 depicts the process of recognition phase.

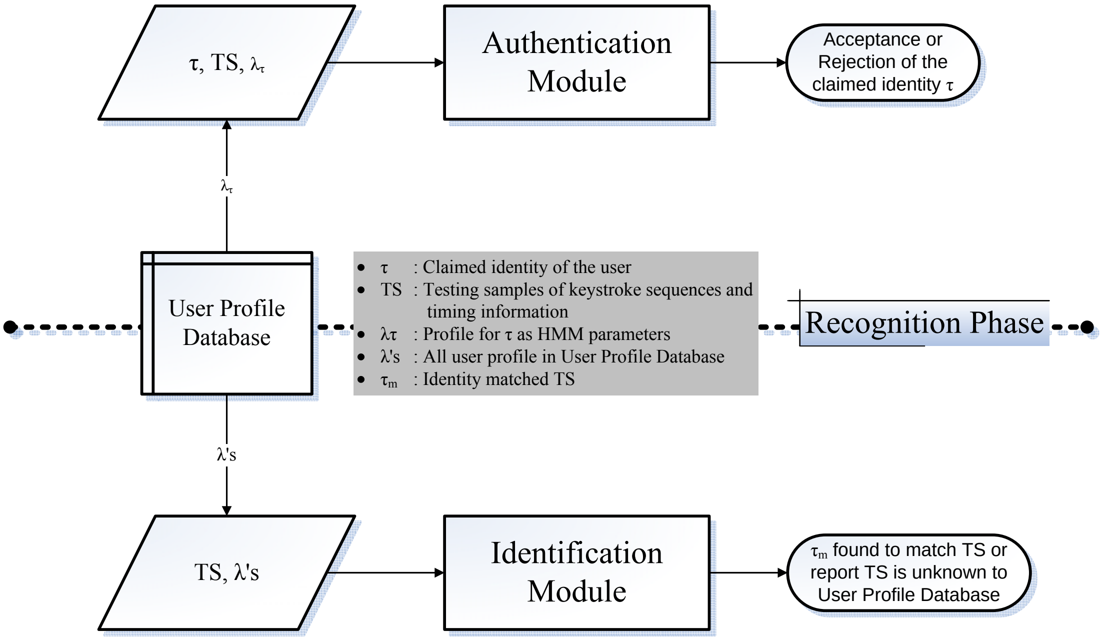

τ, TS, λ_τ
Authentication Module
Acceptance or Rejection of the claimed identity τ
λ_τ
User Profile Database
• τ : Claimed identity of the user
• TS : Testing samples of keystroke sequences and timing information
• $\lambda\tau$: Profile for τ as HMM parameters
• λ's : All user profile in User Profile Database
• τ_m : Identity matched TS
Recognition Phase
λ's
TS, λ's
Identification Module
τ_m found to match TS or report TS is unknown to User Profile Database

Figure 3.6: Flow chart of recognition phase for fixed-text keystroke analysis

3.8. Authentication Strategy

The target strings to be analyzed in traditional login-password authentication mechanism are username and password. We can use two strategies as follows:

- ♦ O-Strategy

  The claimed identity is accepted if both username and password passed the verification phase. This strategy requires users make no mistakes on both target strings.

- ♦ A-Strategy

  The claimed identity is rejected if both username and password denied at the recognition phase. This strategy allows users to make almost most one mistake on one of the target strings.

4. Experiments and Results

4.1. Experiment Setting

The experiment was conducted via web browser. A client-side JavaScript is used to gather the timing information of keystroke. Parts of the volunteers are colleagues and alumni of NCTU. Other parts of volunteers were anonymous from Internet. User provided their login name and password via html form, just like the way commonly employed by the web-based application. The timing accuracy we used is 1 millisecond. In this experiment, we use digraph as the segment size of keystroke sequence.

4.2. Data Collection

For the collection of reference samples, 58 volunteers provided two familiar strings as login name and password for 20 times. As to the collection of testing samples, the above 58 volunteer tried to be authenticated in their own account as legitimate users for 15 times, 870 testing samples were used to evaluate FRR. Another 257 anonymous volunteers tried to be authenticated in legitimate users' accounts. Each account was attacked between 44 and 82 times. Total 3528 imposter testing samples were collected.

The lengths of login name and password are between 4 and 14. Figure 4.1 show the distribution of target string length.

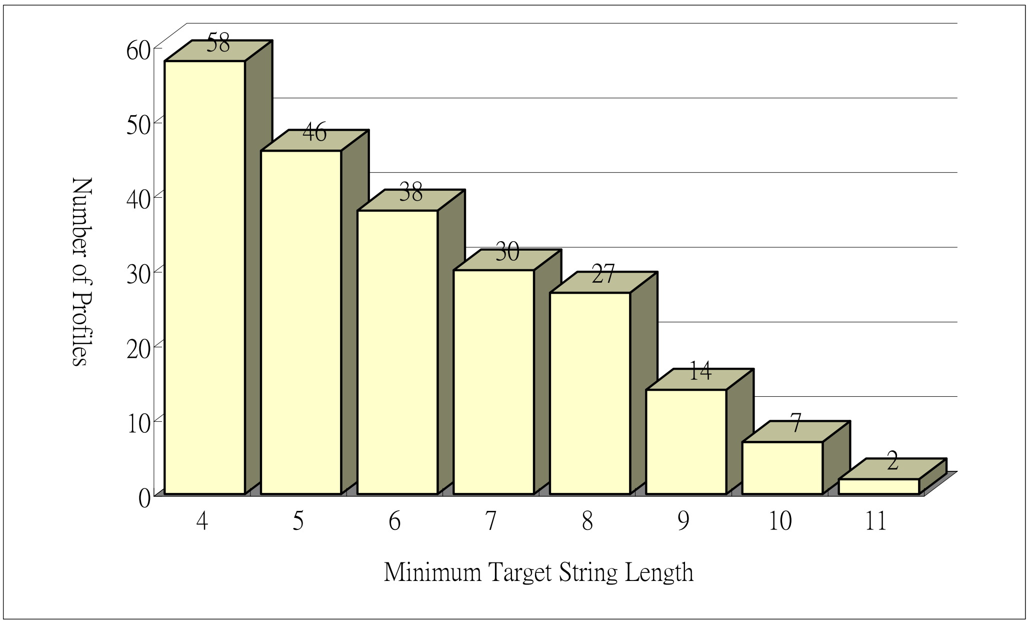


Figure 4.1: Target string length distribution of reference samples

4.3. Evaluation

We evaluate the value of standard deviation weighting factor ε between 0.2 and 3.5 with interval of 0.1 for both strategy. Figure 4.2 to Figure 4.5 shows the FAR and FRR of O-Strategy with minimum target length of 9, reference sample size of 5, 10, 15, 20, and 35 possible standard deviation weighting factor.

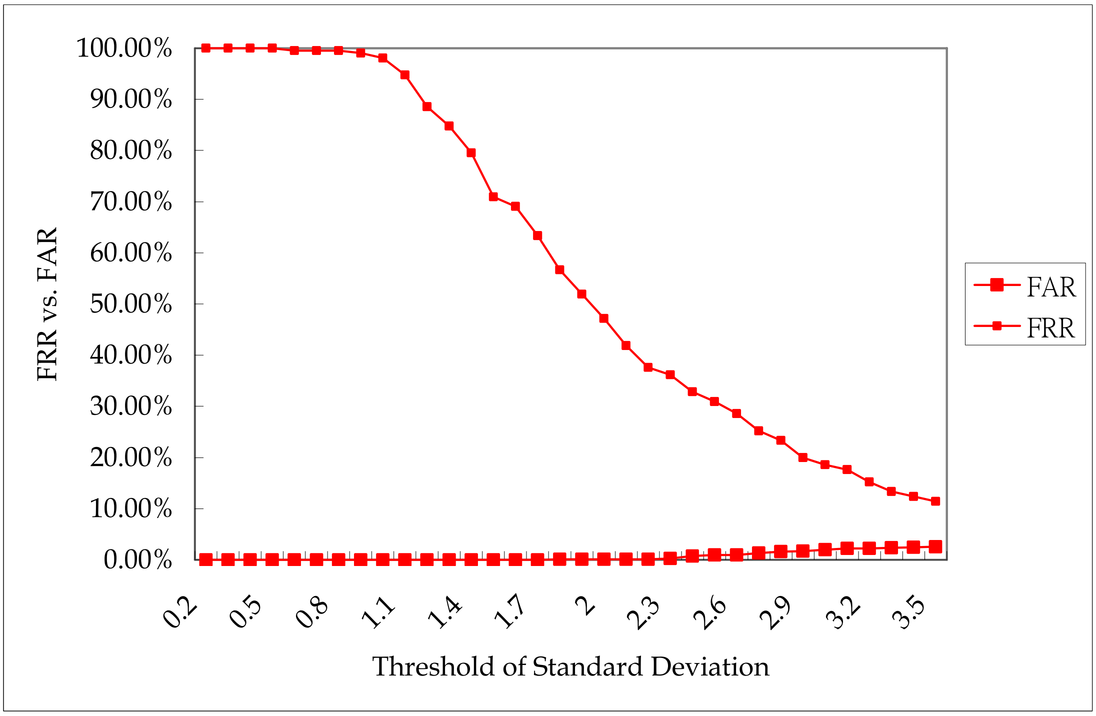


Figure 4.2: O-Strategy - Minimum target string length = 9, reference sample size = 5

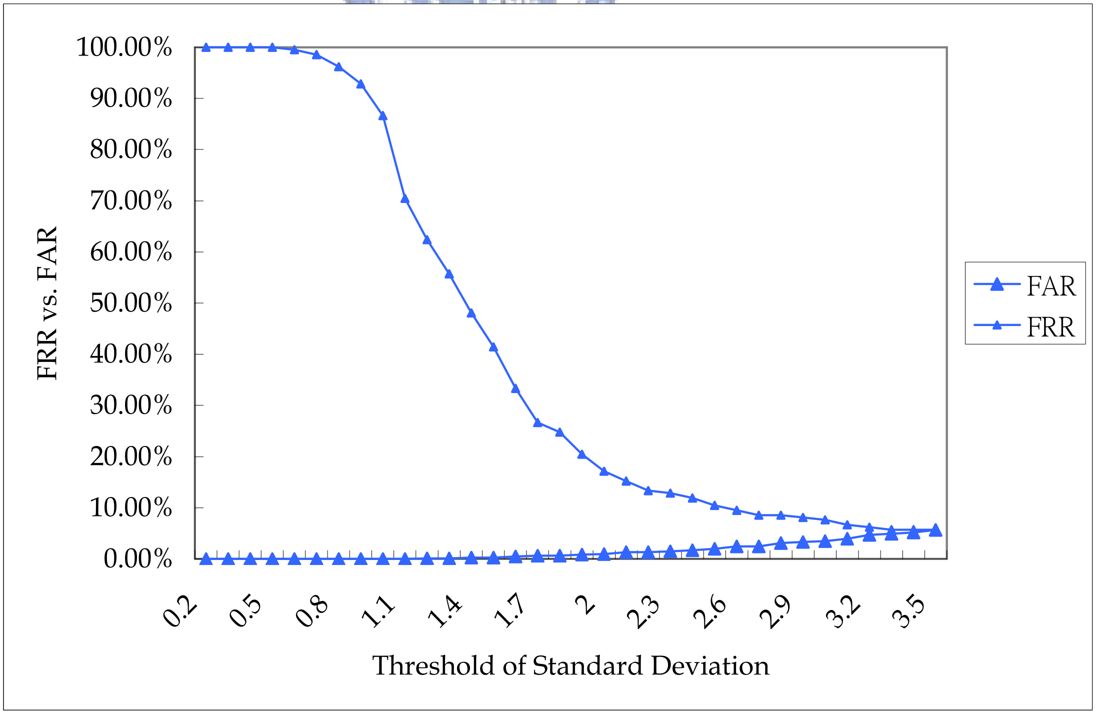


Figure 4.3: O-Strategy - Minimum target string length = 9, reference sample size

= 10, EER = 5.71%

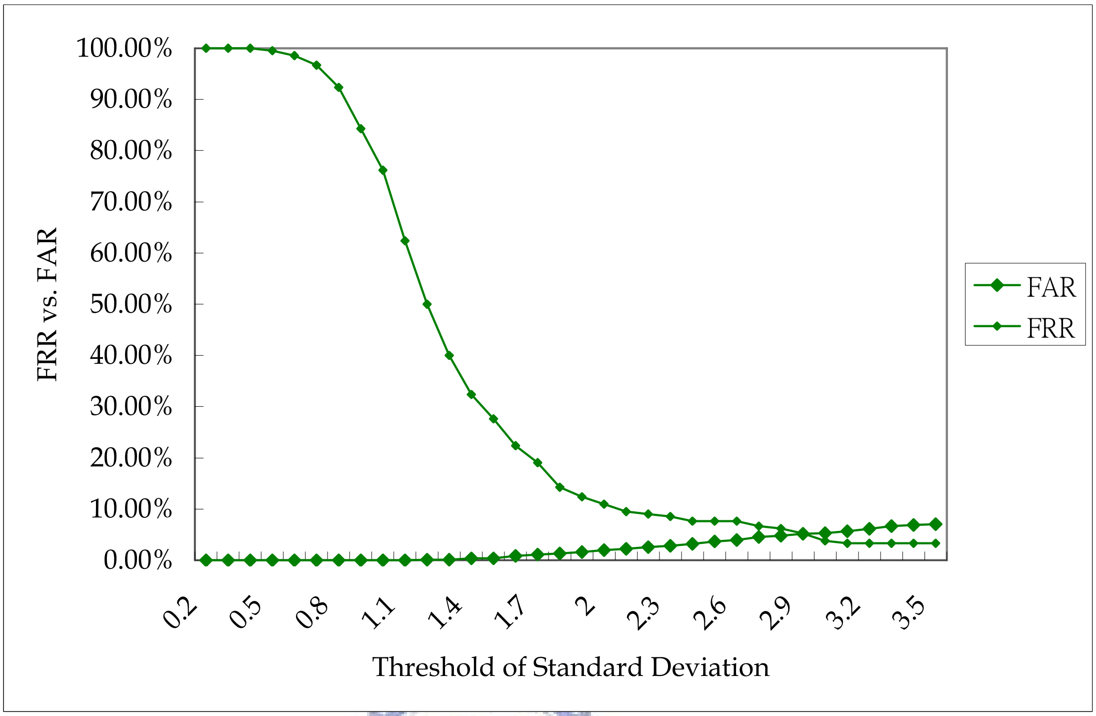


Figure 4.4: O-Strategy - Minimum target string length = 9, reference sample size = 15, EER = 5.24%

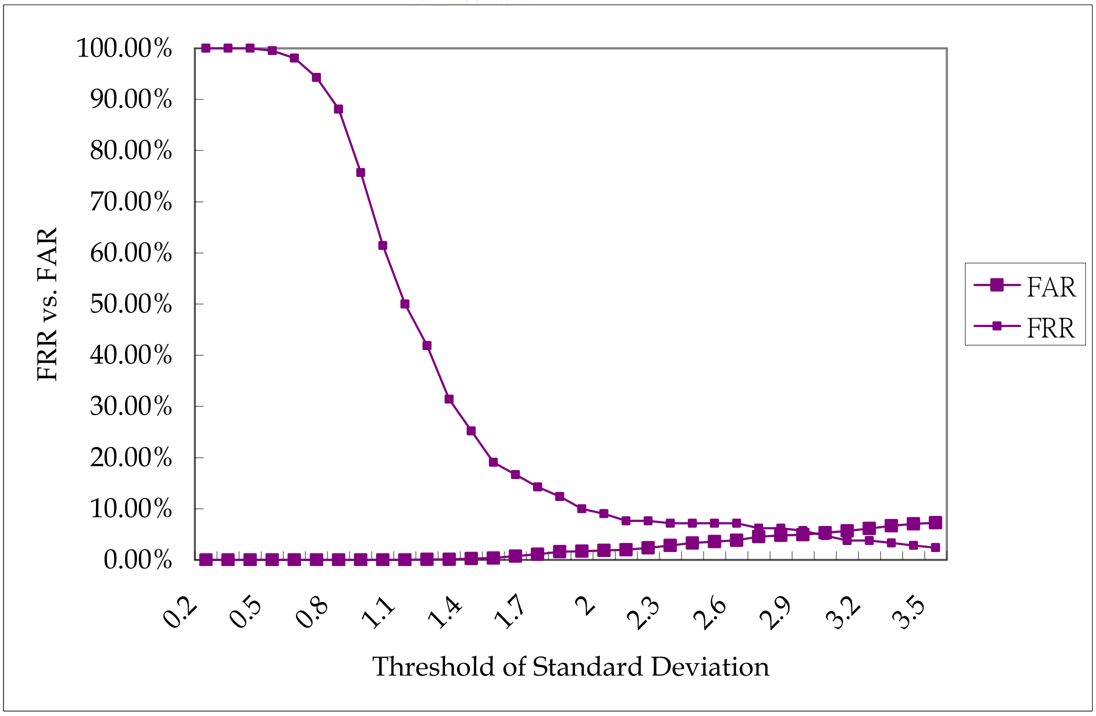


Figure 4.5: O-Strategy - Minimum target string length = 9, reference sample size = 20, EER = 4.76%

Figure 4.6 to Figure 4.9 shows the FAR and FRR of A-Strategy with minimum target length of 9, reference sample size of 5, 10, 15, 20, and 35 possible standard deviation weighting factor.

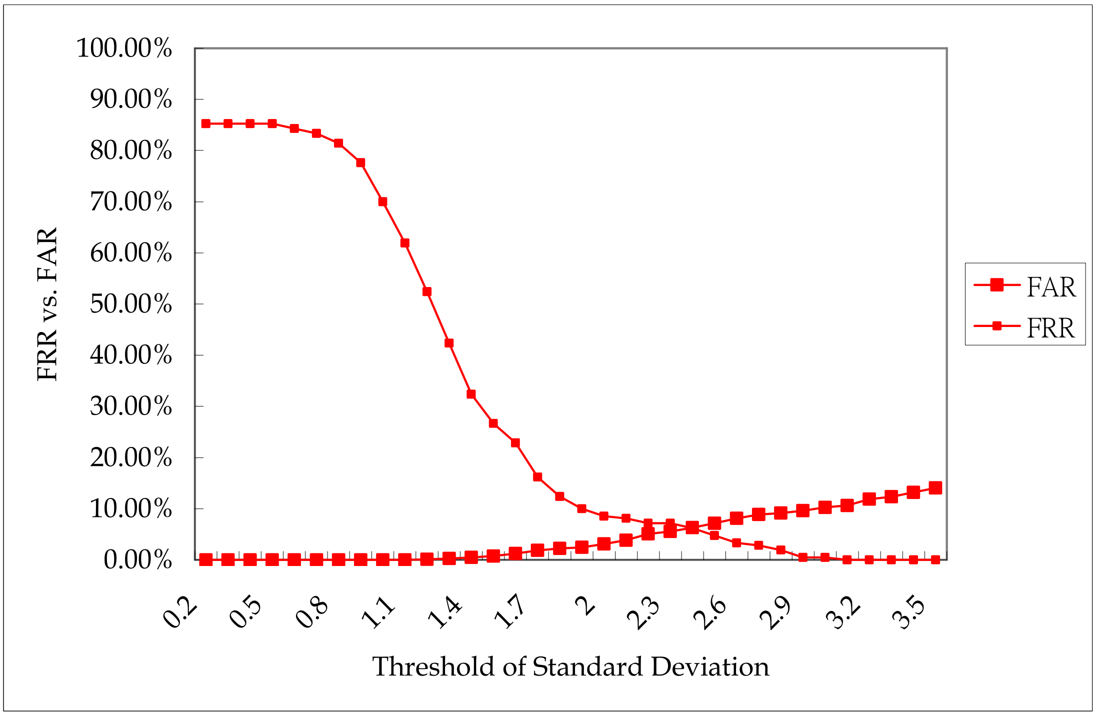


Figure 4.6: A-Strategy - Minimum target string length = 9, reference sample size = 5, EER = 6.19%

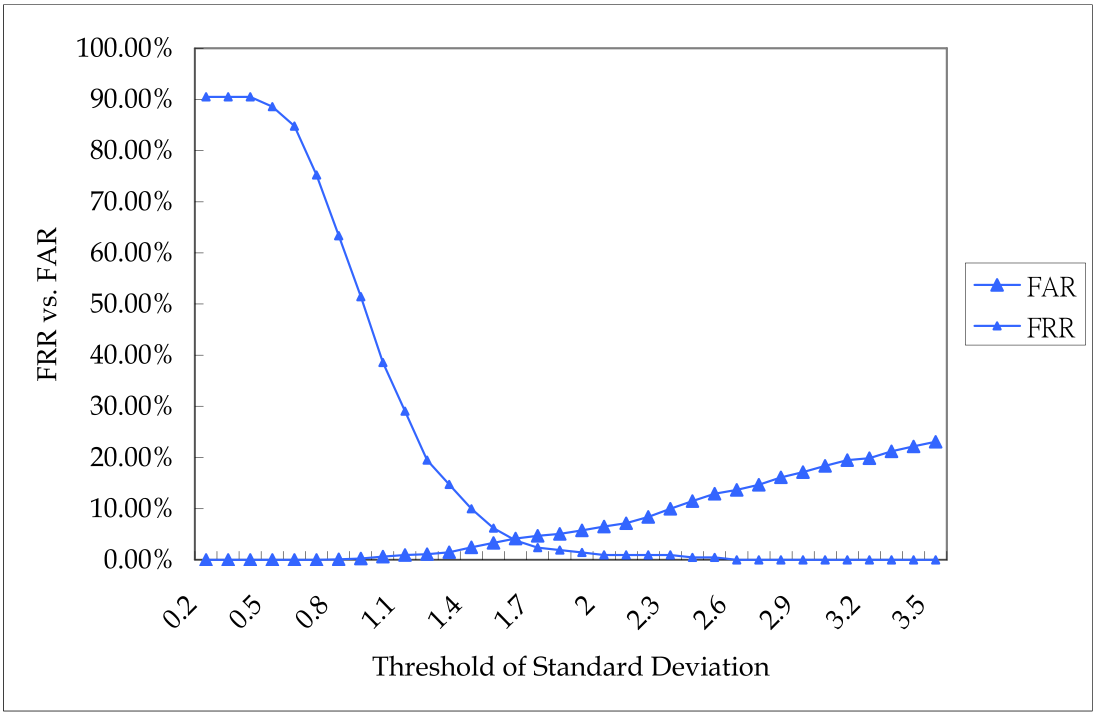


Figure 4.7: A-Strategy - Minimum target string length = 9, reference sample size = 10, EER =3.81%

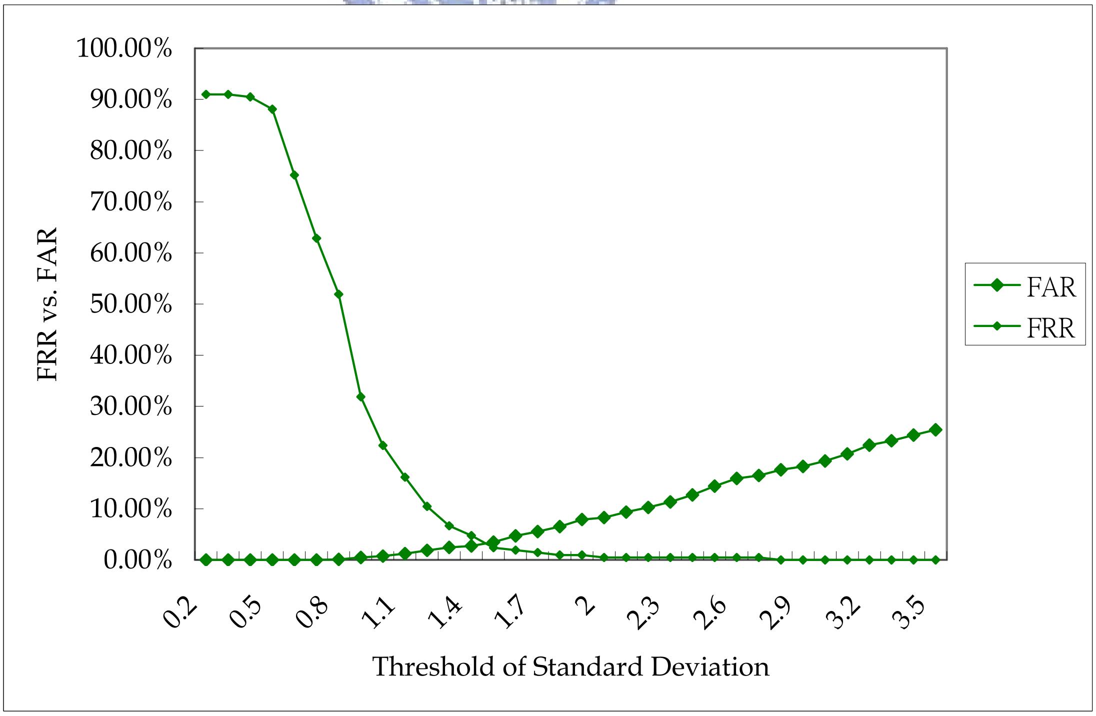


Figure 4.8: A-Strategy - Minimum target string length = 9, reference sample size

= 15, EER =2.91%

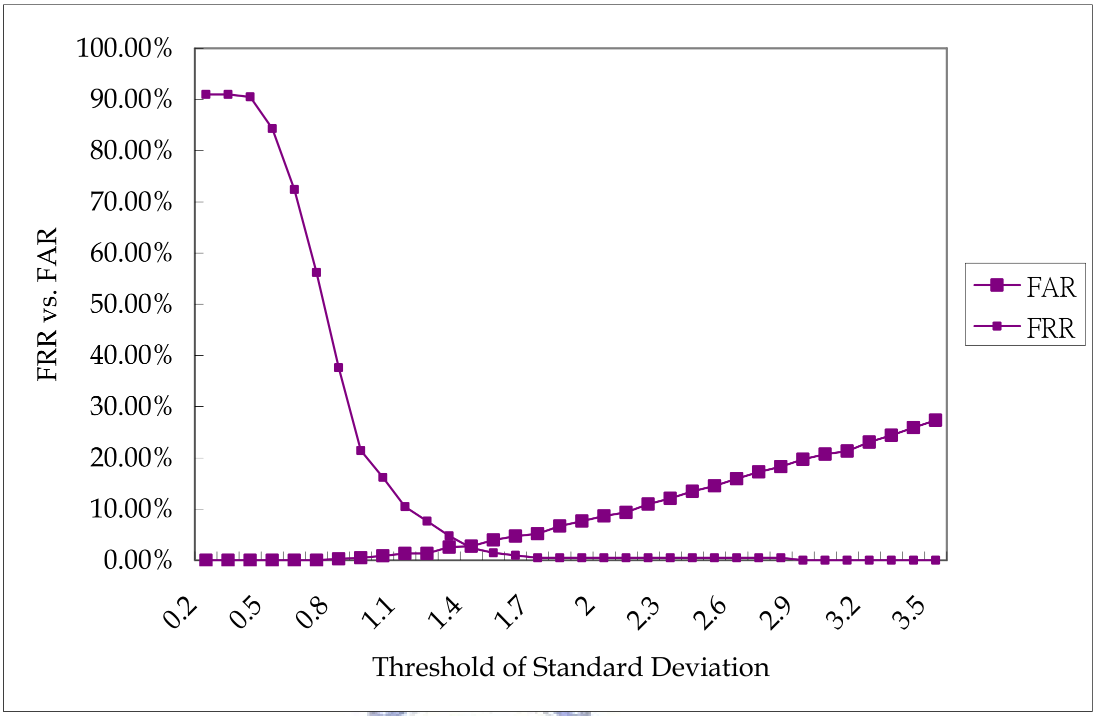


Figure 4.9: A-Strategy - Minimum target string length = 9, reference sample size = 20, EER =2.54%

We can see from Figure 4.2 to Figure 4.9 that A-Strategy obtained better ERR than O-Strategy. In Figure 4.10 and Figure 4.11, we show that the relation of EER vs. minimum target string length.

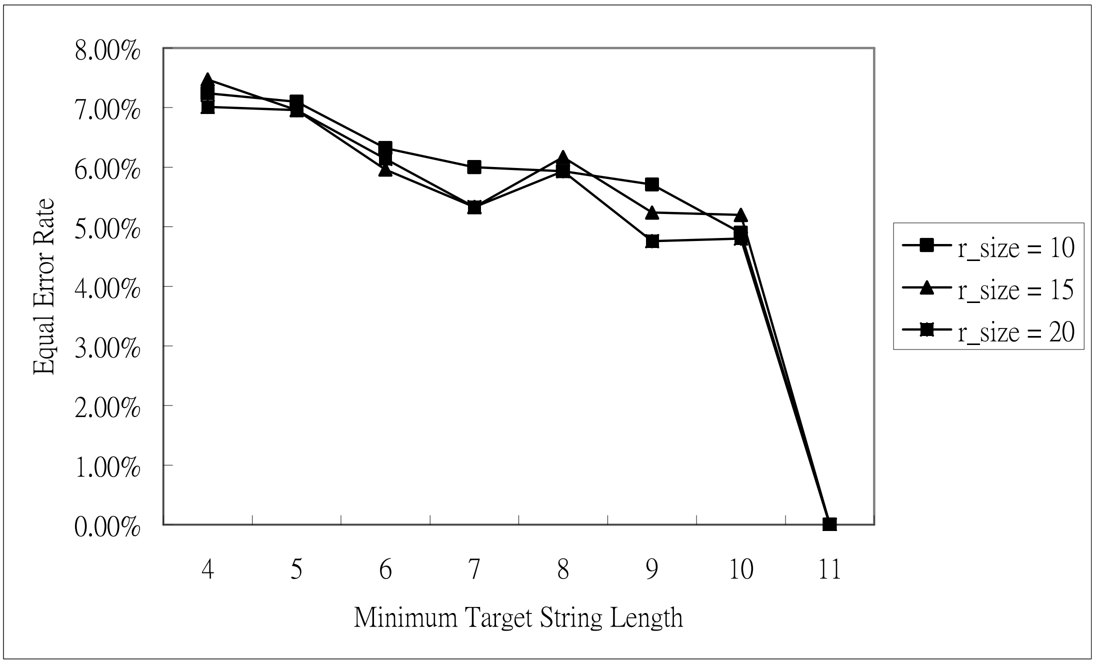


Figure 4.10: O-Strategy – EER vs. Minimum target string length

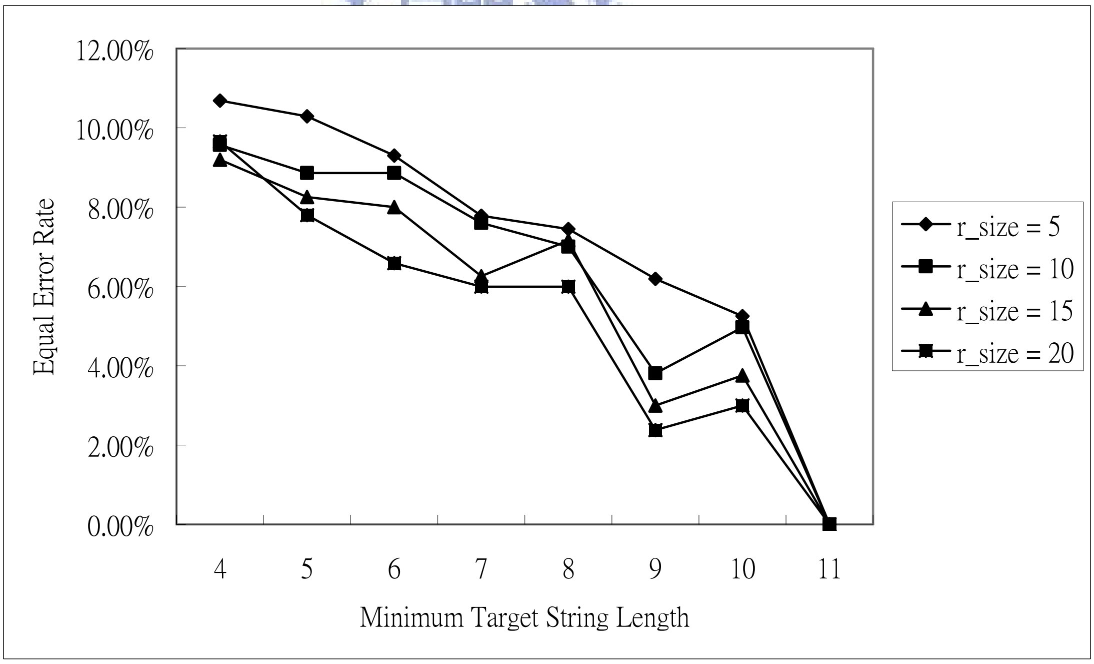


Figure 4.11: A-Strategy – EER vs. Minimum target string length

We can see that as the EER drops as the minimum target string length increases.

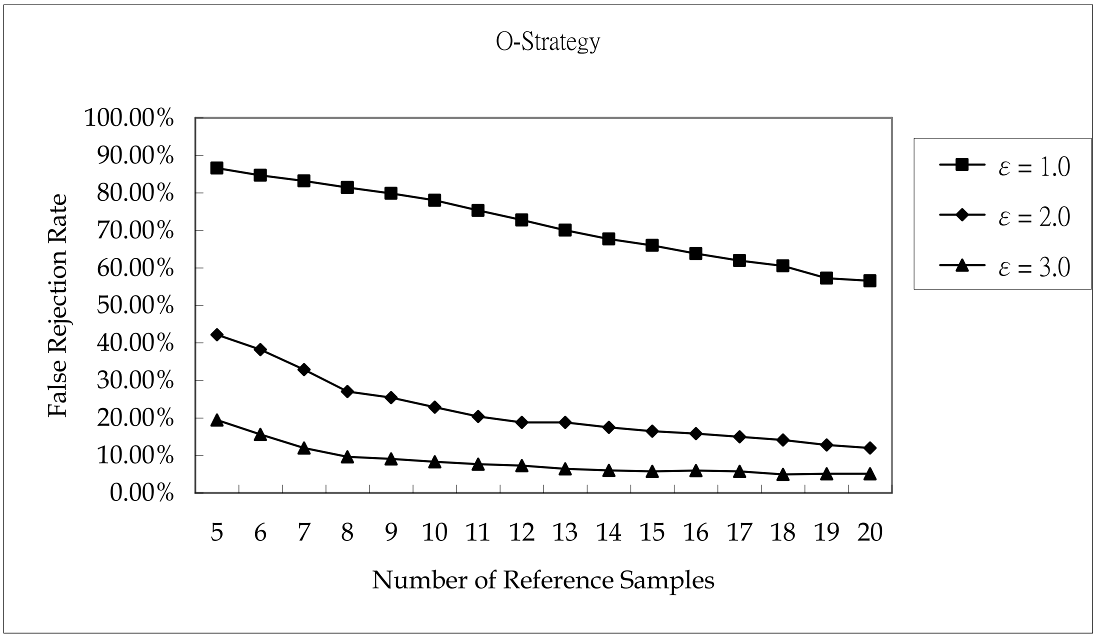


Figure 4.12: O-Strategy: FRR with different number of reference samples

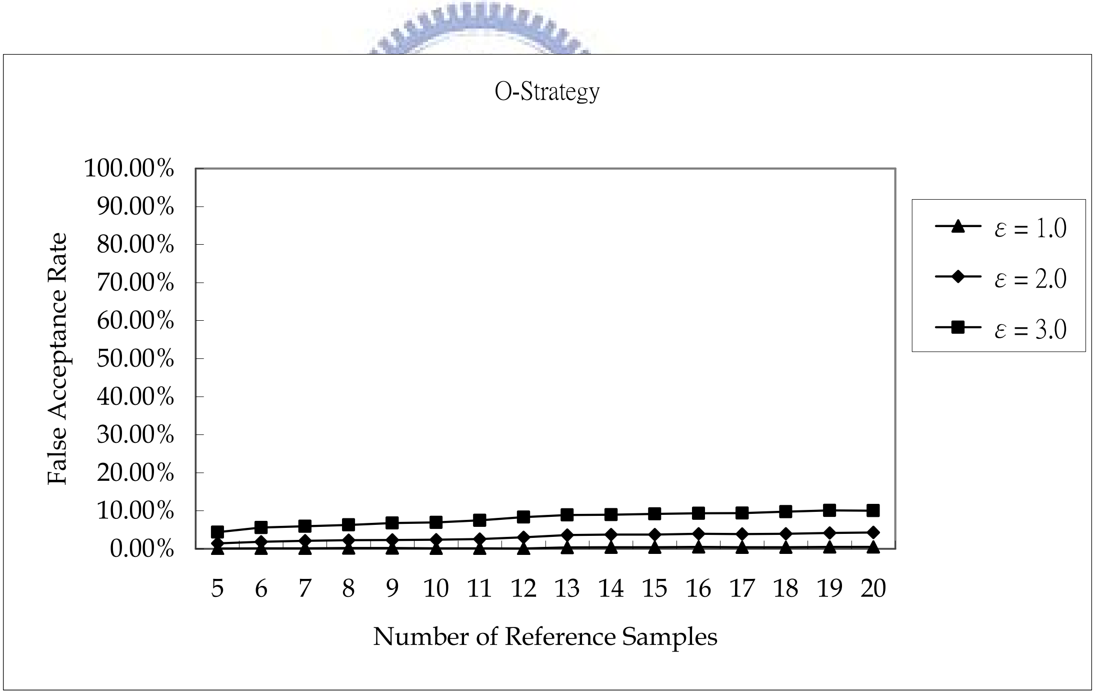


Figure 4.13: O-Strategy: FAR with different number of reference samples

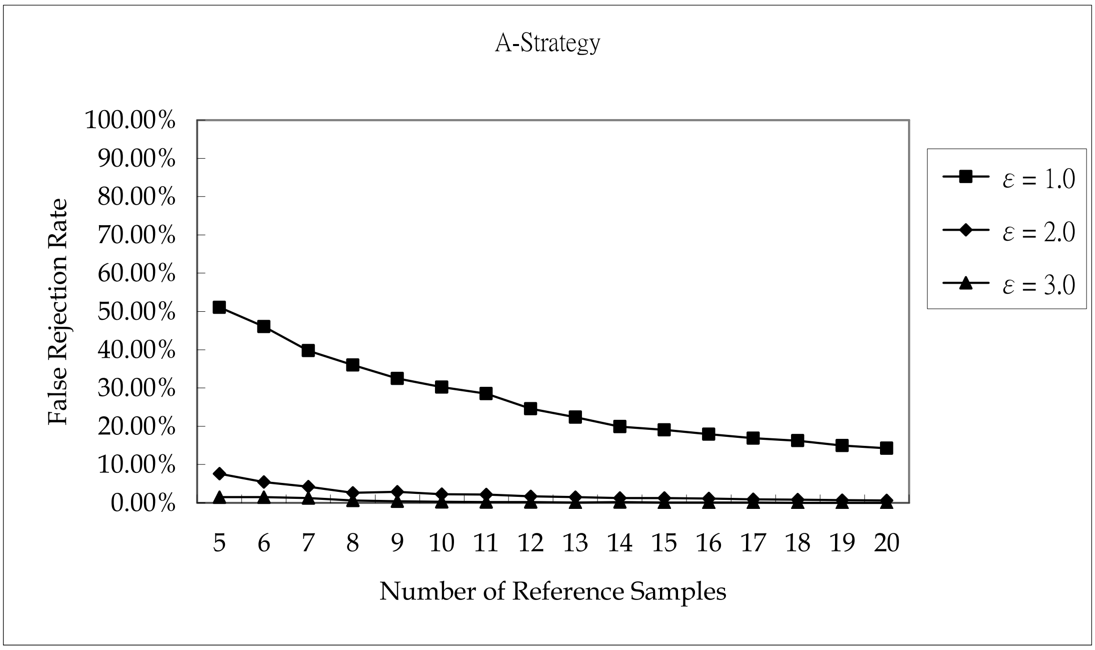


Figure 4.14: A-Strategy: FRR with different number of reference samples

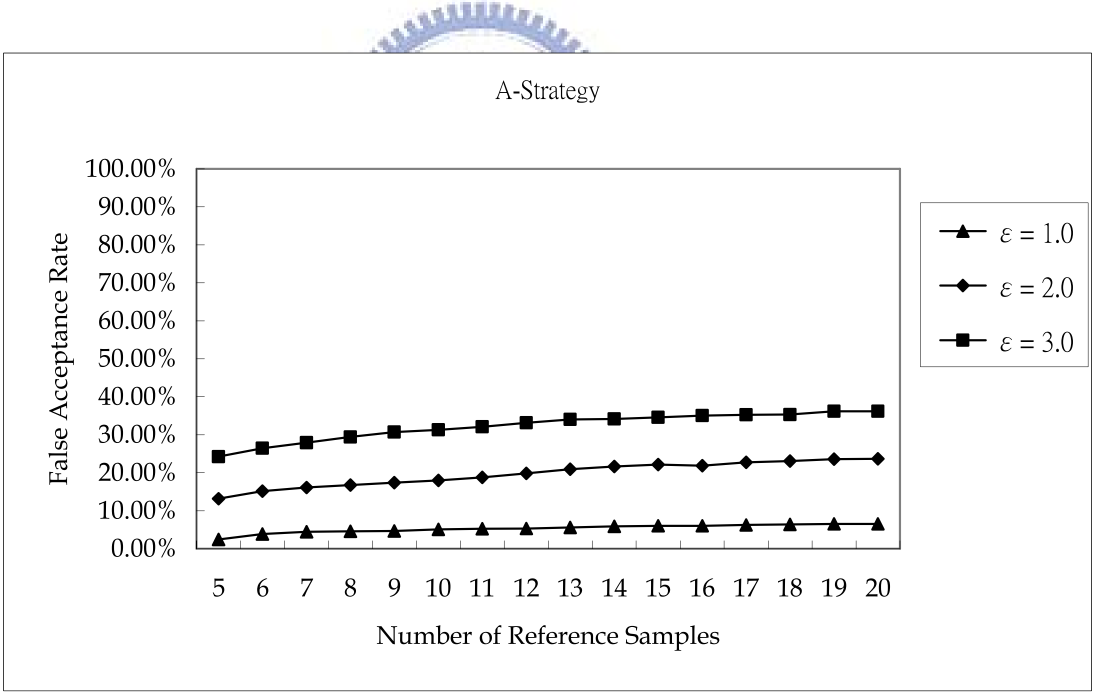


Figure 4.15: A-Strategy: FAR with different number of reference samples

We can see from Figure 4.12 to Figure 4.15 that FRR of both strategies drop as the number of reference samples increase, and FAR of both strategies lift slightly as the number of reference samples increase.

5. Conclusions

Our approach achieved the EER of 2.54%, which is near 2 % - values generally considered to be acceptable for this type of system. The ERR of our scheme can be improved as we conduct more experiment to collect more reference sample with length longer than 10.

As to future work, we can combine the proposed scheme with the analysis of the surfing route to the login page. The proposed model can be extended to devise scheme for free-text keystroke analysis, such as continuously real-time identity verification.

6. References

[1] D. Gunetti and C. Picardi, “Keystroke Analysis of Free Text”, ACM Transactions on Information and System Security (TISSEC), vol. 8, no. 3, pp. 312-347, Aug 2005.

[2] L. C. F. Araujo, L. H. R. Sucupira Jr., M. G. Lizarraga, L. L. Ling, and J. B. T. Uabu-Tti, “User Authentication Through Typing Biometrics Features”, IEEE Transactions on Signal Processing, vol. 53, no. 2, pp. 851-855, Feb. 2005.

[3] S. T. Magalhaes, K. Revett, and H. M. D. Santos, “Password Secured Sites – Stepping Forward with Keystroke Dynamics”, Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP’05), pp. 293-298, Aug. 2005.

[4] W. G. de Ru and J. H. P. Eloff, “Enhanced Password Authentication through fuzzy logic,” IEEE Expert, vol. 17, no. 6, pp. 38–45, Nov. 1997.

[5] K. Revett and A. Khan, “Enhancing Login Security Using Keystroke hardening and Keyboard Gridding”, Proceedings of the IADIS MCCSIS, 2005.

[6] S. T. Magalhaes, H. M. D. Santos, “An Improved Statistical Keystroke Dynamics Algorithm”, Proceedings of the IADIS MCCSIS, 2005.

[7] A. Peacock, X. Ke, and M. Wilkerson, “Typing Patterns: A Key to User Identification”, IEEE Security & Privacy, vol. 2, no. 5, pp. 40-47, Sep 2004.

[8] J. Leggett and G. Williams, “Verifying Identity via Keystroke Characteristics”, International Journal of Man-Machine Studies, vol. 28, no. 1, pp. 67-76, 1988.

[9] S. Haidar, A. Abbas, and A. K. Zaidi, “A multi-technique approach for user identification through keystroke dynamics,” in Proc. IEEE International Conference on Systems, Man, and Cybernetics, vol. 2, pp. 1336–1341, 2000.

[10] D. Song, P. Venable, and A. Perrig, “User Recognition by Keystroke Latency Pattern Analysis”, Apr. 1997.

[11] F. Monrose and A. Rubin, “Authentication via Keystroke Dynamics”, Proceedings of the 4th ACM conference on Computer and Communication Security, pp. 48-56, Apr. 1997.

[12] M. I. Jordan, “An Introduction to Probabilistic Graphical Models”. In preparation.

[13] D. X. Song, D. Wagner, and X. Tian, “Timing Analysis of Keystrokes and Timing Attacks on SSH”, In 10th USENIX Security Symposium,

pp.337-352, Aug. 2001.

[14] S. Russell and P. Norvig, “Artificial Intelligence, A Modern Approach”, Prentice Hall, 1995.

[15] P. Dowland, H. Singh, and S. M. Furnell, “A Preliminary Investigation of User Authentication using Continuous Keystroke Analysis”, Proceedings of 8th IFIP Annual Working Conference on Information Security Management and Small System Security, Sep. 2001.

[16] P. Dowland, H. Singh, and S. M. Furnell, “Keystroke Analysis as a Method of Advanced User Authentication and Response”, Proceedings of the IFIP TC11 17th International Conference on Information Security: Vision and Perspectives, pp. 215-226, May. 2002.

[17] F. Bergadano, D. Gunetti, and C. Picardi, “User Authentication through Keystroke Dynamics”, ACM Transactions on Information and System Security (TISSEC), vol. 5, no. 4, pp. 367-397, Nov 2002.

[18] R. S. Gaines, W. Lisowski, S.J. Press, and N. Shapiro, “Authentication by Keystroke Timing: Some Preliminary Results”, Rand Report R-256-NSF. Rand Corporation, 1980.

[19] R. Joyce and G. Gupta, “Identity Authentication Based on Keystroke Latencies”, Communication s of the ACM, vol. 33, no. 2, pp. 168–176, 1990.

[20] D. Umphress and G. Williams, “Identity Verification through Keyboard Characteristics”, International Journal of Man-Machine Studies, vol. 23, no. 3, pp. 263-273, 1985.

[21] A. J. Mansfield and J. L. Wayman, “Best Practices in Testing and Reporting Performance of Biometric Devices”, Biometrics Working Group, Aug. 2002.

[22] L. R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, Proceedings of the IEEE, vol. 77, No. 2, Feb. 1989.

[23] S. Bleha, C. Slivinsky, and B. Hussien, “Computer-Access Security Systems Using Keystroke Dynamics”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 12, Dec. 1990.