Practical Application of Keystroke Dynamics Authentication

Yuankai Guo,* Myungho Jung,† Junwei Shi,‡ and Xu Wang§
University of Utah

I. INTRODUCTION

Password authentication has been one of the most popular method to identify. It is ubiquitous for systems such as Web sites, SSH, and mobile phones. However, attackers' techniques to steal password have also been developed in many ways. Therefore, we need to collect additional information to figure out identity other than just plain text of passwords. For example, access from different IP or MAC address can be suspected as an attack. However, it makes users inconvenient if they are travelling or buy a new device. Keystroke dynamics can be extracted while typing password as well as used as identity verification[3]. The existing users don't have to buy an additional device. Only a small change in the server will be required to apply this mechanism.

II. RELATED WORKS

Many algorithms to detect outliers were studied and test on performance and precision[5]. The various algorithms were used for pattern matching, such as neural network, Support Vector Machines, K-Nearest Neighbors, and Euclidean distance. For most researches, hundreds of patterns from dozens of people were collected and tested. Unlike those studies which tested on practical experiments, we focused on the variation of the users. One of the defects of the result from most existing tests is that they intentionally collected data from people in a short period of time. If the group only consists of experienced users, the data would be biased. Also, the variation of patterns cannot be observed in a short period of data. In some cases, a test in well-controlled environment can produce more reliable result than practical experiments.

III. THREAT MODEL

Password-based authentication is ubiquitous for identification on the internet. However, as the number of web services has dramatically increased, it becomes impossible for people to remember tons of passwords and most of them are using the same password for all web sites. It will be a problem if a cracker breaks in a insecure server and steals users' passwords, the password can be used for other web services as well. Therefore, it will be useless even if a web site is protected by almost perfect secure system.

Unlike the password, keystroke dynamics is one of the biometric authentication such as face, fingerprint, and iris recognition[4]. Thus, it changes as time goes as well as it can be stored as a data structure which is appropriate to compare similarity but it is difficult to imitate the pattern. Also, the method can be applied on the existing password authentication system as an additional layer. It will make the authentication system more secure and robust.

IV. METHOD

A. Data structure

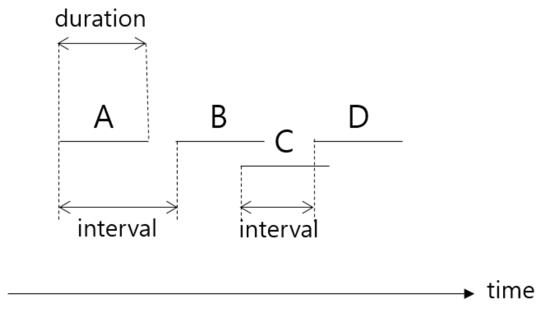

FIG. 1. Timing measurement in interval and duration

The timing vector of keystroke patterns can be measured as intervals between keys and keystroke durations[2]. Therefore, a pattern of $n+1$ numbers will be created from typing n characters password including return key. In most studies, the vector is measure like this; however, as seen in Figure 1, we applied it in a slightly different way. The interval in our project measured not from previous release time but from previous press time. By doing so, there is no negative value and storage can be saved by store only unsigned data. The timing vector can be considered as a point in $n+1$ dimensional space. Then, we can measure the distance of two points, and apply machine learning or neuron network algorithm to detect novelties.

The pattern needs to be normalized because the storage is limited and in order to equivalent comparison among users. First, the timing values are divided by the maximum value and stored with the maximum value.

* u0942867@utah.edu
† myungho.jung@utah.edu
‡ sjwarthas2009@gmail.com
§ xu724@cs.utah.edu

Then, the values becomes less than 1 and the data size can be controlled by floating point precision. Also, two different length of passwords cannot be evaluated with the same error bound. Therefore, the sum of values should be divided by a factor of dimension of the point. Or if the computation is not obvious, we can multiply the error bound by $\sqrt{d}$ such that d is the order of dimension.

V. ALGORITHM

A. Novelty Detection

To filter imposter's patterns, a novelty detection algorithm is required. There are many researches to detect anomaly. The algorithms can be divided into statistical type, neural network, pattern recognition based on learning, and heuristics[1]. We tested three common algorithms, which are the Nearest Neighbor, Mahalanobis distance, and one class SVMs, with changing the patterns.

VI. IMPLEMENTATION

A. Encryption

It's not secure to transmit the password and pattern in plaintext through the network. So we need to protect the password and pattern by encryption or message digest. In our implementation, we use one handshake protocol instead of multiple handshakes for better efficiency. The format of authentication message is KeyHash(TS|Pattern)|TS|Pattern such that TS is timestamp. The Key is 128bit generated from MD5 message digest of password. We generate Hash(TS|Pattern) using SHA-1, which provides integrity protection for the message. We encrypt the Hash(TS|Pattern)|TS|Pattern by AES algorithm. Moreover, we add a salt to for better diversity. With the timestamp and salt, it's invulnerable to rainbow attack.

How to prevent replay attack? In our implementation, the server record the timestamp. If the server receives a message with the same timestamp which already exists in the time window, the server will refuse the message. Of course, the server will also drop the message with invalid timestamp.

However, there is some drawback in the protocol. Since the client couldn't authenticate the identity of the server, it's vulnerable to in-the-middle attack. Besides, the protocol is not secure when it comes to server break-in attack. The attacker could also modify the time in the server.

VII. ANALYSIS

We created a pattern generator to test algorithms. Our tests are focused on the prediction of the variation of user's patterns and comparing performances of pattern matching methods. The error rates for pattern detection are divided into two parts which are False Acceptance Rate(FAR) and False Rejection Rate(FRR)[2]. For each test case, the two error rates are computed for comparing performance.

A. Environment

Test program is coded in python using Scipy, Numpy, and scikit-learn library. For each test, 1000 evenly distributed patterns of imposters are generated because we want to simulate the environment that a lot of imposters try to imitate the pattern. Also, the same number of patterns of genuine users are generated and the distribution of points are the Gaussian. It should be similar to a user's patterns.

Our tests are focused on the variation of the patterns over time. First, only an interval of patterns would be fluctuated. Suppose that the password is mixed with a sequence of alphabets and that of numbers like 'abcd1234.' In this cases, the user's patterns of word and numbers may be stable except the interval between them. Second, we assume that the user's patterns would be fast or slow; however, those would be proportional. The third assumption is that the user may use different types of keyboards such as laptop, desktop keyboard, and touch keyboard on mobile devices. Then, we expected that the patterns will form multiple clusters.

1. *Test Results*

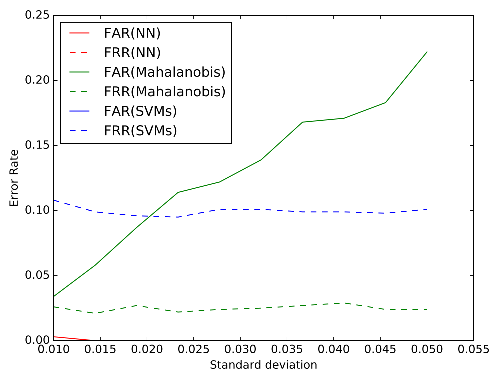

FIG. 2. Test result of patterns that only one of the intervals varies

Most of the error rates are not visible because they are very close to 0. As seen in the three tests, we concluded

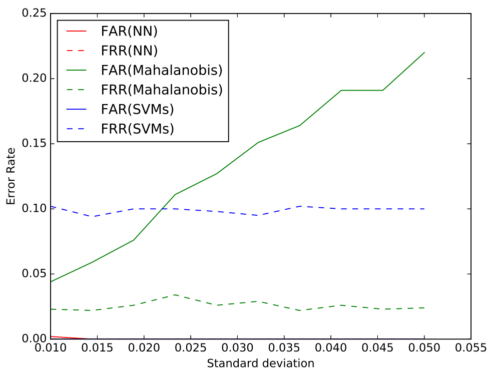

FIG. 3. Test result of proportional patterns

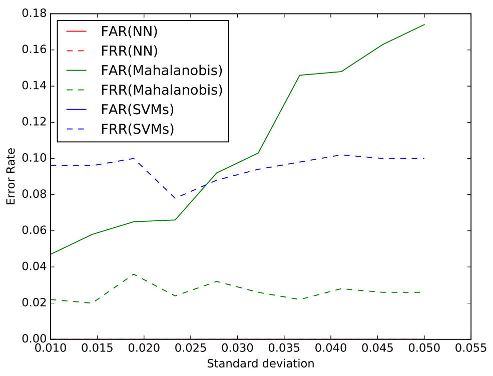

FIG. 4. Test result of patterns of multiple clusters

that the performances of the three algorithms are not much different regardless of the variation of the patterns. The most accurate method among them was the nearest neighbor method, the SVMs was next, and the Mahalanobis distance showed the worst performance. And with respect to computation time, the SVMs was the fastest, the nearest neighbor was next, and the Mahalanobis distance was the slowest. However, it would be the difference in library used. The SVMs and nearest neighbor using ball tree was implemented using scikit-learn library. On the other hand the Mahalanobis distance was measured using scipy library.

VIII. DEFECTS

Although there are many researches and practical application of Keyboard Dynamics, there are problems in authenticating with the method. Assume that the mechanism is applied for Web-based system. Then, people can move cursors using arrow keys or mouse while typing and editing password. We just ignored the cases and assumed that the web page allows only password typed at once. Though it would make the users inconvenient, the web service will become more secure. Also, key stroke patterns may be able to be recorded by spyware with key logger. In general, we don't have to worry about the attack because the pattern is encrypted before sent. If the encryption is impossible in the system, it can be protected by the attack by sharing a nonce with the server and adding it to the pattern. Finally, a user's patterns change more frequently and radically than other physiological traits or behavioral characteristics, and thus it would be a problem. We suggested a system that evaluates only the latest patterns like sliding window.

IX. CONCLUSION

Through the project, we checked the algorithm is working by demo program and also tested the performance between algorithms. We suggested that the possible web-based authentication using keystroke dynamics by implementing and testing actual login web page. Unfortunately, it is not sure that the test data is significant compared to that from other research. However, the application developed for the experiments will be useful to test other algorithms with a myriad of possible fixed patterns and variations. It will help to create various environments for experiments.

[1] Salil P Banerjee and Damon L Woodard. Biometric authentication and identification using keystroke dynamics: A survey. *Journal of Pattern Recognition Research*, 7(1):116–139, 2012.

[2] Sungzoon Cho, Chigeun Han, Dae Hee Han, and Hyung-Il Kim. Web-based keystroke dynamics identity verification using neural network. *Journal of organizational computing and electronic commerce*, 10(4):295–307, 2000.

[3] R Stockton Gaines, William Lisowski, S James Press, and Norman Shapiro. Authentication by keystroke timing: Some preliminary results. Technical report, DTIC Document, 1980.

[4] M Karnan, M Akila, and N Krishnaraj. Biometric personal authentication using keystroke dynamics: A review. *Applied Soft Computing*, 11(2):1565–1573, 2011.

[5] D Shanmugapriya and Ganapathi Padmavathi. A survey of biometric keystroke dynamics: Approaches, security and challenges. *arXiv preprint arXiv:0910.0817*, 2009.