VFA: A Variable-Factor Authentication Framework for Mobile Users

Kai Chen[1,2], Weifeng Chen[3,*], Zhen Xu[1], Dongdai Lin[1], and Yazhe Wang[1]
[1] State Key Laboratory of Information Security, Institute of Information Engineering Chinese Academy of Sciences, Beijing 100093, China
[2] University of Chinese Academy of Science, Beijing 100049, China
[3] Department of Math, Computer Science and Information Systems, California University of Pennsylvania, 250 University Ave, California, PA 15419
Email: {chenk, xuzhen, ddlin, wangyazhe}@iie.ac.cn; chen@calu.edu

Abstract —Multi-factor authentication (MFA) has been widely used in various scenarios. By combining multiple forms of authentication, MFA effectively provides security assurance. Due to the rapid developments of mobile devices, especially smart phones, more and more sensitive information is now stored or accessible on smart phones. How to protect smart phones' security is now more important than ever. Unfortunately, because of the special features of smart phones such as computational limitations and input constraints, existing MFA schemes could not be directly used on smart phones. In this paper, we propose a new concept of Variable-Factor Authentication (VFA) for smart phones. VFA dynamically adjusts the number of authentication factors based on whether a user is suspicious or not. We implement a prototype to exam the performance. The experiment results show that, compared to MFA, VFA provides significant convenience to legitimate users whereas maintain the security protection to suspicious users.

Index Terms—Multi-factor authentication, variable authentication factors, local outlier probabilities

I. Introduction

Today, almost every Internet service requires a username and a password to protect sensitive information. The combination of a username and a password is easy to be implemented on the server. Although recently many services require a complex password that includes special characters, password-based authentication schemes are still not strong enough for them. For example, the TeraGrid stakkato incident [1], [2] demonstrates that password-based authentication schemes can be easily exploited by attackers and always have a widespread impact.

More and more Internet services including online banking (e.g., Chase.com) and investment accounts (e.g., Vanguard .com, Tiaa-cref.org) now implement multi-factor authentication mechanisms (MFA). In addition to a username and a password, a user also needs to input another piece of confidential information, such as a PIN or the answer to a secret question. The benefits of using these stronger authentication methods are obvious. The University of Tennessee's National Institute for Computational Science (NICS), who has provided resources to the TeraGrid, has promoted the use of MFA since its founding. Its system log shows that a number of individual user accounts have been compromised across the TeraGrid in recent years, among which, none was due to a stolen password [3].

A. Multi-Factor Authentication (MFA)

The most common MFA scheme combines a token (e.g., a physical card) and a secret (e.g., PIN), implemented in most bank ATM cards. A recent study on two-factor authentication based on smart cards and passwords was presented by Yang *et al*. [4]. Combination of passwords or tokens with biometrics is another favorable authentication method, such as Biohashing [5], [6] or PalmHashing [7]. Recently, three-factor authentication [8] has been introduced to incorporate the advantages of the authentication based on password, token, and biometrics.

Obviously, the combination of multiple authentication factors can improve the security of authentication. But MFA incorporates not only the advantages but also the disadvantages of each factor, resulting in a trade-off between security and convenience. For example, an additional hardware device (e.g., a smart card) is required when users log into systems. This trade-off is more important to mobile users. Due to the developments of smart phones, more and more apps have become available for mobile phones. Some of them contain sensitive information, such as PayPal apps or email clients. Mobile users need the stronger authentication methods (e.g., MFA) to protect their sensitive information. But, due to the computational limitations and input constraints of the mobile phones, such type of authentication methods is always perceived as a barrier to usability. A convenient and secure authentication scheme is desirable for mobile phones. Unfortunately, there are no existing MFA methods that meet these requirements for mobile users.

Manuscript received February 12, 2015; revised June 24, 2015.
This work was supported by the "Strategic Priority Research Program" of the Chinese Academy of Sciences (No. XDA06010701), "Special expenses of scientific apparatus and equipment" of State Key Laboratory of Information Security (No. Y4D0031302) and National Natural Science Foundation of China (Grant No.61202476).
Corresponding author email: chen@calu.edu, chenk@iie.ac.cn.
doi:10.12720/jcm.10.6.366-379

B. Variable-Factor Authentication (VFA)

In this paper, we propose a generic framework for multi-factor authentication on mobile devices. We refer to this new framework as the Variable-Factor Authentication framework (VFA). VFA aims to provide secure authentication and convenience to mobile users, by dynamically adjusting the number of authentication factors. Mobile users tend to take their mobile devices along with them all the day and access the mobile devices anytime and anywhere. Thus, behaviors of mobile users are easier to form some patterns. A survey conducted by ourselves (Section III) showed that, when using some special applications, more than half of participants form some usage patterns. Does this mean that the usage patterns can be used to distinguish the attackers from the legitimate users directly? The answer is no, because the process of identifying the user based on the user's behaviors cannot entirely avoid false positive errors and false negative errors. But, the usage pattern can be used to find anyone suspicious. This helps VFA to dynamically adjust the number of authentication factors, providing convenience to mobile users while achieving secure authentication.

The idea of VFA is quite straightforward. First, VFA builds a reference model (or called a user model) of user behaviors through a training process. After the reference model is built, VFA compares the user's current login behavior to the reference model. If the user is considered to be a suspicious user (i.e., the user's behavior does not match the reference model), VFA will maximize the number of authentication factors to provide secure authentication; otherwise, VFA minimizes the number of authentication factors to provide convenience.

VFA includes three stages and combines different techniques, including density-based clustering and abnormal detection. The three stages are the pre-processing stage, the user model building stage and the variable-factor authentication stage. In the pre-processing stage, all authentication factors are divided into two categories: mandatory factors and optional factors. For example, mandatory factors could include a simple 4-digit passcode, whereas optional factors could include a graphical password or a voice recognition. In the second stage: user model building stage, VFA enables both the mandatory-factor authentication and the optional-factor authentication for every login request. For every successful login (i.e., a user passes both the mandatory-factor authentication and the optional-factor authentication), VFA collects multiple sensor signals (e.g., accelerometer) from the mobile device and records the login data (e.g., login time) during the successful login. At the end of this stage, VFA builds a reference model based on the collected signals and login data, using a simplified density-based clustering algorithm. In the third stage: the variable-factor authentication stage, VFA enables the mandatory-factor authentication for each login request. After the user passes the mandatory-factor authentication, various signals and login data (such as login time, location, login interval, sensor signal) are extracted and compared to the reference model. If the user is considered as a suspicious user, the optional-factor authentication is enabled subsequently. Otherwise, for a legitimate user, no more optional-factor authentication is needed.

C. Contributions

In summary, this paper has the following contributions.

a) First, we introduce the new concept of "variable-factor authentication (VFA)" to protect the mobile applications. VFA dynamically adjusts the number of authentication factors in runtime environment based on the user's behaviors.
b) Second, we design a generic framework that converts the existing MFA schemes into the VFA schemes. Different techniques are proposed in the design, including density-based clustering algorithm and abnormal detection. Density-based clustering algorithms are used to build the reference model using the collected signals and login data. Abnormal detection techniques are used to determine whether or not a user is suspicious.
c) Third, we implement a VFA prototype on Android phones, the most widely used mobile phones, to exam the performance. Our experiments show that, for a legitimate user, VFA effectively provides convenience by reducing the number of authentication factors.

The rest of the paper is organized as follows. Section II describes previous efforts on multi-factor authentication for mobile devices. Section III motivates our work through a mobile user study and establishes the threat model for our framework. The proposed VFA framework is presented in Section IV. Implementation and experiment evaluation are addressed in Section V. Finally we conclude our paper in Section VI.

II. Related Work

Several multi-factor authentication schemes for mobile users have been proposed in recent years. We introduce some of them in this section.

In addition to the password, some Web services require the user to answer to a secret question if the user's location changes from the last login (e.g., QQ.com). But the accuracy of this type of authentication mechanisms is insufficient. Even if the login location changes, the login request may still come from a legitimate user. Moreover, these Web services get the users' location via their IP address, resulting in big error.

Greenstadt and Beal [10] introduced the notion of cognitive security on computers and other personal devices. They proposed a combination of multiple low-fidelity authentication factors to produce an ongoing positive recognition of a user. Shi *et al.* [11]-[13]

proposed implicit authentication, which was viewed as a step forward in the direction of realizing the vision described in [10]. Implicit authentication used observations of user behaviors (SMS, phone calls, browser history and location) for authentication. The authors mentioned that this approach could be used as a second factor to augment password-based authentication to achieve higher-assurance authentication in a cost-effective and user-friendly manner, or be used to replace passwords to relieve users from the burden of entering complicate passwords. Chow *et al*. [14] extended implicit authentication to cloud computing. They presented an authentication framework, which was based on implicit authentication, for mobile users in the cloud. But the user model used by implicit authentication is only the combination of multiple probability density functions. It did not remove noises from sample data. Moreover, as we mentioned in Section I, the schemes that directly identifying the user based on the user's behaviors cannot completely avoid false positive errors and false negative errors, due to the nature of probability functions.

Some previous efforts have focused on how to make authentication mechanisms more efficient and convenient for mobile users. Multi-factor authentication schemes have the potential to improve security but face usability problems. Czeskis *et al*. [15] proposed PhoneAuth, which was essentially a two-factor authentication scheme but offering the same authentication experience as traditional passwords alone. They used mobile devices as the second authentication factor to provide cryptographic identity assertions. But it may not be the case in all scenarios. PhoneAuth only applies to users who have a suitable mobile phone with them and attempt logins from a browser on another device (e.g., desktop or laptop).

Progressive authentication proposed by Riva *et al*. [16] is an attractive solution for users who do not use any security mechanism on their devices. Progressive authentication intends to balance security and convenience by reducing the number of times a user is requested to authenticate. Its key insight is to combine multiple signals (biometric, continuity, possession) to determinate the user's security level. Progressive authentication is a single-factor authentication scheme whose main goal is to protect important applications (e.g., Android apps or iOS apps) from unauthorized use. If the user's security level is high enough, no authentication is required. Otherwise, the user must go through the authentication.

III. Motivations and Assumptions

Compared to desktops and laptops, mobile devices have different usability and various limitations. First, mobile users normally take their mobile devices with them all day. They can access their mobile devices anytime and anywhere. Thus behaviors of mobile users using special applications tend to follow some patterns. In the next section, we conduct a survey to confirm this idea. Second, the computational limitations and input constraints make it difficult for complex authentication mechanisms to be deployed on mobile devices. With all these considerations, we define in this section the motivation, basic assumptions and the threat model of VFA.

A. Usage Patterns

As we mentioned previously, VFA dynamically adjusts the number of authentication factors, based on whether a user is suspicious or not. Whether a user is suspicious is determined based on user's behaviors. Thus we conducted a survey to look at the behaviors of users using some special applications. Are there some patterns of user behaviors?

In our survey, we chose eight most frequently used applications as target. These eight target applications include the Weixin client, a game application (i.e., Plants vs. Zombies), the Weibo client, the default music player, the E-mail client, the calendar, the Taobao client and the Baidu client. As shown in the Appendix, we asked the participants in our survey to answer the questionnaire. For each target application, three questions were asked:

a) Whether do you use this application?
b) If you use the application, do you think that you have formed some usage patterns?
c) If you think that you have some patterns, please tell us where, when and how you use the applications.

For question 3, we gave some check boxes, e.g., “Using during commute hours”, “Using after supper on the bed” and so on. Here, we explain the meaning of the options. For example, the option, “Using during commute hours”, means that: “over a relatively fixed time period”, “along a relatively fixed route”, “single-handed operation” and “the phone being in portrait mode”. Similar to “Using during commute hours”, each option in the question 3 represents some usage patterns.

A total of 75 mobile users participated in our survey. The mobile devices of the participants were primarily Android phones and iPhones. 31 of the 75 participants were male and 44 are female. Our survey covered different age groups: 65.3% of participants in the age range of 20 to 30 years, 29.3% of participants in the age range of 30 to 40 years and 5.3% of participants in the age range of 40 to 50 years.

The survey results are presented below:

Weixin, also called WeChat, is a multifunction software for mobile devices. By using Weixin, users can chat with others, or share photos with friends. Weixin also helps users to subscribe to news on the Internet. It also provides other functions, such as browsing web sites, calling a taxi, mobile payment and so on. It had grown rapidly in recent years. The 71 participants in our survey installed Weixin on their mobile devices. As shown in Fig. 1, 23.9% of participants logged into Weixin after wake up, and 38% of participants were used to launching Weixin during commuting between their homes and offices. In summary, 74.6%=(100%-25.4% of no pattern)

of participants thought that they had formed some usage patterns.

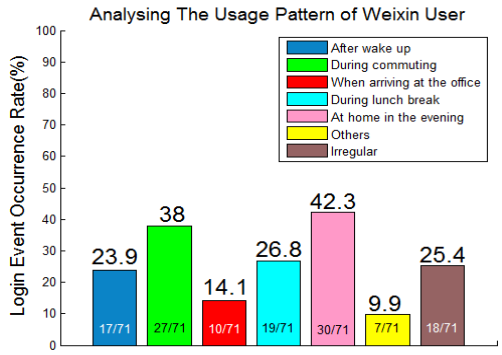

Fig. 1. Survey for weixin users.

66.7% of participants (i.e., 50 participants) in our survey played games in their mobile phones. As illustrated in Fig. 2, the survey result showed that the ratios of participants who played mobile games on their way to work and at home in the evening were relatively large, with a percentage of 64% and 52%, respectively. In summary, 84% of participants thought that they had their own habits of gaming.

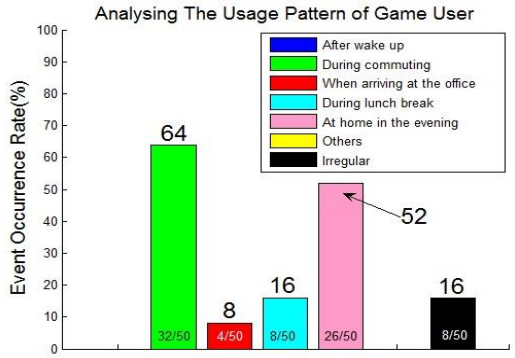

Fig. 2. Survey for game users.

In our survey, the 52 participants listened music by using their phones. Fig. 3 shows the statistic results for the music player users. Among them, 69.2% of participants listened music during commuting between their homes and offices, and 23.1% of participants enjoyed music at home in the evening. In summary, 88.5% of participants thought that they listened music following predictable patterns.

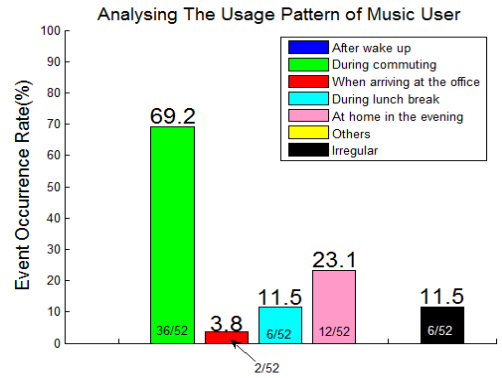

Fig. 3. Survey for music users.

Weibo is a popular MicroBlog application. It allows users to exchange small elements of content such as short sentences, individual images or video links. 90.7% of participants in our survey used Weibo to share their daily lives. Fig. 4 shows the survey results. The amount of participants who logged into Weibo on their way to work and at home in the evening was relatively large. The former was 23.5%. And the latter was 39.7%. The rest of participants were uniformly distributed across others patterns. In summary, 61.8% of participants thought that they had their own habits of using Weibo application.

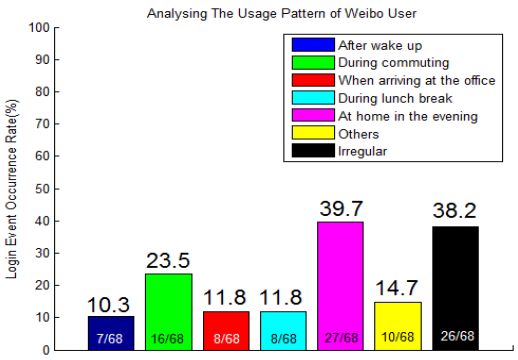

Fig. 4. Survey for weibo users.

Baidu is a popular search engine in China. In our survey, the 60 participants were used to searching through Baidu. Fig. 5 is the survey result for the Baidu client. For instance, 33.3% of participants were used to search information at home after Supper. In summary, 46.7% of participants thought that they formed their own usage patterns.

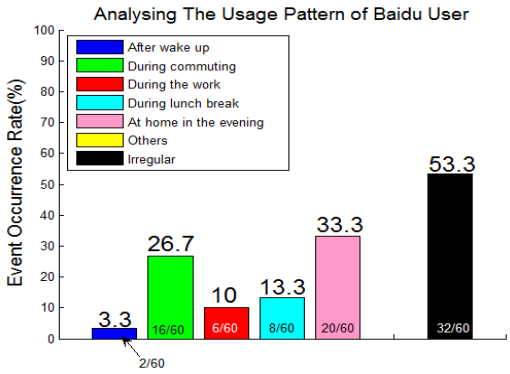

Fig. 5. Survey for baidu users.

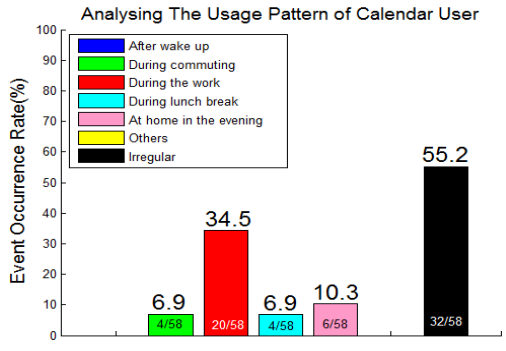

Fig. 6. Survey for calendar users.

According to the survey result, the calendar application plays an important role in most participants' work. 77.3% of participants (i.e., 58 participants) in our survey used the calendar every day. As shown in Fig. 6, 44.9% of participants thought that they used the calendar application following some patterns.

Most of participants (totally, 73 participants) received or sent emails during their work time. As shown in Fig. 7, only 28.8% of participants thought that they processed their emails without any patterns. In another word, 71.2% of participants thought they followed some patterns when using the email client.

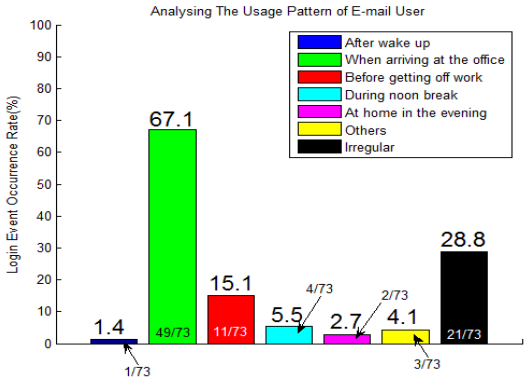

Fig. 7. Survey for E-mail users.

The Taobao client is an e-commerce application, belonged to Alibaba. Most of young people like shopping using it. In our survey, 89.3% of participants (i.e., 67 participants) used it. Fig. 8 shows the survey results for the Taobao client. In summary, about 50.7% of participants thought they had some patterns when shopping.

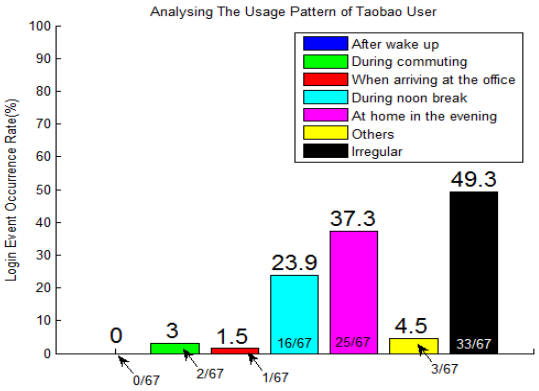

Fig. 8. Survey for taobao users.

TABLE I: SURVEY RESULTS

Application Name	*Ratio of participants whose behaviors followed some patterns*
Weixin	74.6%
Game	84%
Music Player	88.5%
Weibo	61.8%
Baidu	46.7%
Calendar	44.9%
E-mail	71.2%
Taobao	50.7%

As shown in Table I, in average, 65.3% of the participants thought that they had formed some patterns when using the target applications. Specially, for some applications (e.g., the Weixin and the game application), the ratio of participants whose behaviors follow usage patterns is very high. The survey results prove the feasibility of the idea that whether a user is suspicious can be determined based on his behaviors, and also motivate VFA. Comparing MFA, VFA could bring benefits to more than half of the users at least.

B. Assumptions, Threat Model and Goals

VFA has the following two assumptions. As briefly described above, to build a user's reference model, VFA collects multiple sensor signals and log data from mobile devices. So we first assume the availability of low-cost sensors in mobile devices. Actually, the sensors used by VFA have been widely provided in current mobile devices. Second, we assume that the sensor signals used by VFA are accurate and consistent.

The main goal of VFA is to provide convenience while achieving secure authentication at the presence of the following adversaries with different levels of power. The ability of the level 1 adversary is the weakest. The level 3 adversary has the strongest ability.

- Level 1: Adversary possesses none of identity credentials (e.g., password or smart card), and knows nothing about the victim's usage habits.
- Level 2: Adversary possesses the identity credentials of the mandatory-factor authentication. But he or she knows nothing about the victim's usage habits, and does not possess the identity credentials of the optional-factor authentication.
- Level 3: Adversary not only possesses the identity credentials of the mandatory-factor authentication, but also knows the details about the victim's usage habits.

Based on the assumptions and threat model described above, we define the goals of VFA as following:

a) **Make multi-factor authentication schemes more convenient for a legitimate mobile user.** If the number of authentication factors is too many, a legitimate user may feel inconvenient using their phones. Thus the first goal of VFA is to bring convenience to a legitimate mobile user, by reducing the number of authentication factors.

b) **Provide security guarantees with the presence of the level 1 and level 2 adversaries.** The second goal of VFA is to provide security assurances comparable to that of a normal multi-factor authentication scheme, when facing the adversaries with the level 1 and level 2 of power. In other words, VFA could maximize the number of authentication factors for a suspicious user.

c) **Convert existing multi-factor authentication schemes into variable-factor authentication schemes.** The proposed framework should be generic enough to convert an existing multi-factor

authentication scheme to a variable-factor authentication scheme in a short time.

IV. Variable-Factor Authentication

As briefly mentioned in Section I, there are three major stages in VFA: the pre-processing stage, the user model building stage and the variable-factor authentication stage. These three stages are described in more details in this section.

A. Pre-Processing Stage

VFA is a generic framework that converts an existing multi-factor authentication scheme to a variable-factor authentication scheme. The first step for the conversion is to classify authentication factors of the existing multi-factor authentication scheme into two categories: mandatory factors and optional factors. Mandatory-factor authentication will be enabled for every login request, including the request from both legitimate users and suspicious users. Instead, optional-factor authentication is required only for suspicious users.

Mandatory factors should include the most important and fundamental factors. The principles of selecting a mandatory factor include the following.

a) **Convenient**: VFA aims to make multi-factor authentication more convenient for a legitimate user. So complex authentication factors should not be in the mandatory group.
b) **Stronger**: In VFA, mandatory-factor authentication is the first line of defense, so a mandatory factor should not be too weak.
c) **Balanced**: The smaller number of authentication factors that are classified into the mandatory group, the more convenience is provided by VFA for legitimate users. However, a small number of mandatory authentication factors may cause a weak security. We suggest that the number of authentication factors in the mandatory group should be more than half of all authentication factors.

B. User Model Building Stage

In this stage, VFA builds the reference model (namely, the user model) using a simplified density-based clustering algorithm. Building the user model requires sensor data and login data from a legitimate user. To collect these data, VFA enables both the mandatory-factor authentication and the optional-factor authentication for every login request. Users who successfully pass both the mandatory-factor authentication and the optional-factor authentication are assumed to be legitimate, and the sensor data and login data during the successful login process will be collected to build the user model. VFA collects the following types of sensor signals:

a) **Magnetic field sensors**: Magnetic field sensors are measurement instruments used to measure the strength and direction of the magnetic field at a point in space.
b) **Orientation sensors**: Users can use the orientation sensor to determine the position of a device. Orientation sensors provide azimuth, pitch and roll values.
c) **Accelerometers**: A smart phone accesses an accelerometer to measure acceleration. VFA uses the combination of magnetic field sensors, orientation sensors and accelerometers to capture the status of the mobile device when a login request is received. Some users are used to input their account names and passwords in the landscape mode but others prefer to type in the portrait mode.
d) **GPS**: GPS is a space-based satellite navigation system that provides location information. With GPS, VFA can know the location of the user when a login request is received.

The login data recorded by VFA includes the following items:

e) **Login time**: VFA records the time when a login request is received.
f) **Location**: The location information is another key indicator. Besides GPS, wireless location techniques also provide geographic location information. This is because GPS cannot be used in some places, e.g., in underground parking.
g) **Interval**: VFA computes the time interval since last successful login.
h) **Error Rate**: VFA computes the number of failed login requests occurred in the past *t* hours.

At the end of the user model building stage, all the data mentioned above for a legitimate user will be collected to be build the user model, through a three-step process that is described below.

1) Step 1: Standardization

The collected data first needs to be standardized to remove negative impacts of variables that have a large variance. For a data set $X=[x_1,x_2,\cdots,x_n]$, μ presents the mean value of X and σ is the standard deviation of the set. The standardized value $x_i^{'}$ is computed through equation (1).

$$x_i^{'}=\frac{x_i-\mu}{\sigma},(x_i\in X,1\le i\le n) \quad (1)$$

2) Step 2: Projection

Each raw data point collected by VFA consists of three sensor signals (magnetic field, orientation and accelerometers) and four types of login data (login time, location, interval and error rate). Note that the signal from GPS is not included in the raw data point, because it is integrated in the location data. It is difficult to cluster such a raw data point consisting of seven components. Few clustering algorithms can do that. So VFA transforms every raw data point (referred to a high-dimensional data point) into multiple low-dimensional

data points through a multiple-planes orthographic projection, a process to be described next.

Let s_m, s_o and s_a be the signals from the magnetic field sensor, the orientation sensor and the accelerometer, respectively. Let d_t, d_l, d_i and d_e denote the login time, the location, the interval and the error rate, respectively. A raw data point could be presented as a seven-element vector $(s_m, s_o, s_a, d_t, d_l, d_i, d_e)$. VFA chooses "login time" d_t as a reference coordinate and transforms the seven-element vector into six pairs (d_t, s_m), (d_t, s_o), (d_t, s_a), (d_t, d_l), (d_t, d_i), (d_t, d_e). In other words, the high-dimensional data point is projected onto six projection planes, one at a time -- the time-magnetic plane, the time-orientation plane, the time-accelerometer plane, the time-location plane, the time-interval plane and the time-error plane. Taking the time-location plane as an example shown in Fig. 9, time-axis and location-axis form a projection plane. VFA projects orthographically a seven-dimensional data point onto the time-location plane to obtain a two-dimensional data point (d_t, d_l). The same process is applied to other projection planes. So $(s_m, s_o, s_a, d_t, d_l, d_i, d_e)$ is converted as (d_t, s_m), (d_t, s_o), (d_t, s_a), (d_t, d_l), (d_t, d_i), (d_t, d_e). With this approach a seven-dimensional data set is transformed into six two-dimensional data sets. A simplified density-based clustering algorithm is then used to build the user model.

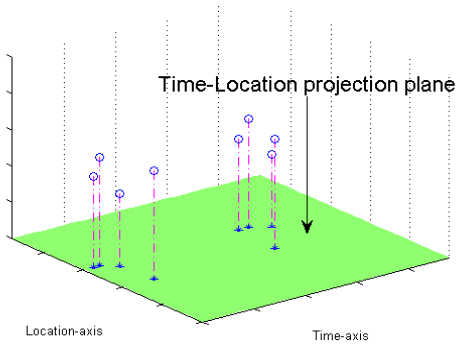

Fig. 9. Project a data point onto the time-location projection plane.

3) Step 3: Building the user model

We propose a simplified density-based clustering algorithm that is applied to each two-dimensional data set obtained in step 2 to build the user model. The user model will consist of six resultant clusters, each from one of the two-dimensional data sets.

Ester *et al.* proposed the fist density-based clustering algorithm -- DBSCAN [17]. Because DBSCAN could identify clusters with arbitrary shapes and specified density, it has caught attention of many researchers. Previous works mainly studied how to discover clusters efficiently [18]-[22]. VFA has different requirements, however. In VFA, clusters do not need to be merged. Instead, VFA focuses on classifying raw data into clusters and removing noises. Thus we use a simplified density-based clustering algorithm.

The simplified clustering algorithm will be applied six times to the six two-dimensional data sets obtained from Step 2 described above. For every data set, the algorithm goes through the following three steps:

a) Construct a grid and distribute all data points in the data set into the cells of the grid.
b) Calculate the density threshold Δ for the grid.
c) Remove noises based on Δ.

Taking the data set (d_t, s_o) as example, each step is described below with more details.

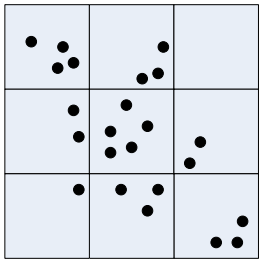

Fig. 10. The grid for (d_t, s_o) and its cells.

a) Each dimension of (d_t, s_o), i.e. the time-axis and the orientation-axis, is divided into m intervals. Hence, the sample space of the data set is partitioned into m^2 rectangle cells. All of these cells form a **grid**. Here, we adopt the method of GDILC [23] to determine the number of intervals m.

$$m = \sqrt{n/\varepsilon} \tag{2}$$

In equation (2), n is the total number of data points in the two-dimensional data set. ε is a coefficient to adjust the value of m, which is set to 15 in VFA, following the recommendation from [23], i.e., $m = \sqrt{n/15}$. As Fig. 15 shows, the data points are distributed into the cells.

b) VFA calculates the density for each cell and the density threshold Δ for the grid. To consider the impacts of neighbor cells when calculating the density of a cell, we use the concept of **shifting cell** introduced in [24]. A shifting cell extends the original cell by 50% in each direction. So if the side of an original cell is L, the side of its shifting cell becomes $2L$. As shown in Fig. 11, the original cell is shown in **solid lines** and its shifting cell is in **dashed lines**. The density of the original cell is defined as the number of data points within its shifting cell. Let $\bar{\delta}$ be the mean value of the calculated densities of all cells in a grid, the density threshold Δ of the grid is obtained through equation (3).

$$\Delta = \begin{cases} 2, & n < 1000 \\ \dfrac{\bar{\delta}}{\log_{10}(n)} \times \rho, & n \geq 1000 \end{cases} \tag{3}$$

where ρ is an adjustable coefficient. Here, we set $\rho = 0.95$, which is the recommended value from [24] to achieve an accurate result.

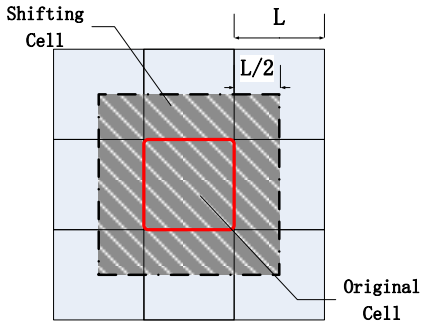

Fig. 11. Shifting grid of a cell.

c) Remove noises from the grid. Cells in the grid that have a density smaller than Δ are considered as noises. All data points in a noise cell are removed from the data set. Each of the remaining cells is represented as a cluster. The resultant clusters become the usage patterns of (d_t, s_o). In the next stage, the abnormal detection will then compare a pending data point to these usage patterns and determine whether the pending data point is suspicious or not.

After these three steps are applied to each of the six two-dimensional data sets, the process of building the user model is finished. The user model consists of the usage patterns in the six data sets.

C. Variable-Factor Authentication Stage

Once the user model is established, VFA enters the variable-factor authentication stage. In this stage, when a login request is received, VFA dynamically adjusts the number of authentication factors, based on the difference between the user model and the signals and data extracted from the current login request. This process is referred to as a **model matching** process.

1) Preliminaries

A local outlier detection scheme is used in VFA for model matching. Here, we first introduce the concept of **local outliers**. Local outlier factor (LOF) [19] is a data mining technique that is proposed for outliers detection [25], [26]. LOF assigns to each object a degree of being an outlier. This degree depends on how isolated the object is with respect to the surrounding neighborhood. But the major problem of LOF is how to interpret the degree in order to decide whether or not the data object is an outlier. LoOP [9], which intends to resolve the problem of LOF, provides an outlier "score" in the range of [0, 1] that is directly interpreted as the probability of the data object being an outlier. The brief computational procedure of LoOP is described below:

a) We assume D being a set of n data objects, and d being a distance function. Let $S \subseteq D$ be a context set of an object $o \in D$. S includes the k-nearest neighbors of o. Breunig [19] suggested that MinPts, namely k, should be at least 10 to remove unwanted statistical fluctuations. Therefore, we chose 11 as the k value in VFA.

b) A standard distance of the objects in S (i.e., the k-nearest neighbors of o) to o can be computed through equation (4).

$$\sigma(o,S) = \sqrt{\frac{\sum_{s \in S} d(o,s)^2}{|S|}} \tag{4}$$

c) The probabilistic distance of o to S is defined as equation (5).

$$\varsigma(\lambda, o, S) := \lambda \cdot \sigma(o,S) \tag{5}$$

In equation (5), $\lambda = \sqrt{2} \cdot \phi^{-1}(\varphi)$, where $\phi()$ denotes the Gaussian error function. The probabilistic distance has the following property:

$$\forall s \in S : prob[d(o,s) \le \varsigma(\lambda, o, S)] \ge \varphi \tag{6}$$

Intuitively, λ is chosen in a way that the sphere around o with radius $\varsigma(\lambda, o, S)$ covers any element in the context set S with probability of φ. Experimentally, the value of λ follows the "three sigma" rule, e.g. $\lambda = 1 \Leftrightarrow \varphi \approx 68\%$, $\lambda = 2 \Leftrightarrow \varphi \approx 95\%$ and $\lambda = 3 \Leftrightarrow \varphi \approx 99.7\%$.

d) The **Probabilistic Local Outlier Factor** (θ) of an object o w.r.t. a context set $S(o) \subseteq D$ is computed through equation (7).

$$\theta_{\lambda,S}(o) = \frac{\varsigma(\lambda, o, S(o))}{E_{s \in S(o)}[\varsigma(\lambda, s, S(s))]} - 1 \tag{7}$$

where $E_{s \in S(o)}[\varsigma(\lambda, s, S(s))]$ is the expected value of $\varsigma(\lambda, s, S(s))$ for all $s \in S(o)$.

e) During the computation of θ, the aggregate value Θ can be obtained through equation (8).

$$\Theta := \lambda \cdot \sqrt{E[\theta^2]} \tag{8}$$

f) Finally, a Gaussian Error Function is applied to obtain a probability value Γ, namely **Local Outlier Probability (LoOP)**, as equation (9) illustrates.

$$\Gamma_S(o) := \max\left\{0, \phi\left(\frac{\theta_{\lambda,S}(o)}{\Theta \cdot \sqrt{2}}\right)\right\} \tag{9}$$

where $\phi()$ again denotes the Gaussian error function.

2) Model matching

Before the model matching procedure can be executed, VFA requires system administrators to assign a weight coefficient to each two-dimensional data set in the user model, and set the probability threshold τ. The sum of these six weight coefficients $(W_m + W_o + W_a + W_l + W_i + W_e)$ is equal to 1. The weight coefficients and the probability threshold can be used to balance between usability and security by the administrators.

The model matching procedure is executed in the following way. VFA enables mandatory-factor authentication for every login request. For every login request, VFA extracts the user's behavior features from the current login request. VFA collects signals $(s_m^{'}, s_o^{'}, s_a^{'})$ from the mobile device and record the login data $(d_t^{'}, d_l^{'}, d_i^{'}, d_e^{'})$. After standardization and projection, each login request results in six pending data points $(d_t^{'}, s_m^{'})$, $(d_t^{'}, s_o^{'})$, $(d_t^{'}, s_a^{'})$, $(d_t^{'}, d_l^{'})$, $(d_t^{'}, d_i^{'})$ and $(d_t^{'}, d_e^{'})$, which will be compared to the user model later.

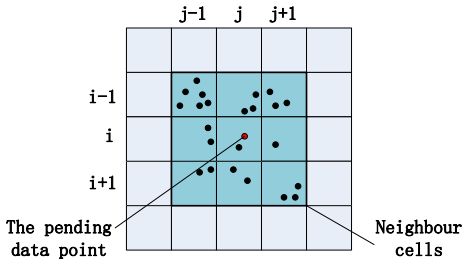

Fig. 12. Neighbor grid cells.

To compare pending data points to the user model, we need to introduce the concept of neighbor cells. As Fig. 12 shows, each pending data point is placed into the corresponding two-dimensional data set in the user model. Assume that the pending data point is placed into the cell $C_{i,j}$. The neighbor cells of the cell $C_{i,j}$ are defined as follows:

Definitioin 1. As illustrated in Fig. 12, cell $C_{i,j}$ and $C_{i^{'},j^{'}}$ are neighbor cells when the following condition is meet.

$$\left|i - i^{'}\right| \le 1, and \left|j - j^{'}\right| \le 1 \tag{10}$$

It means that each grid cell has nine neighbor cells at most.

Next, we present the detailed process of the function **Model-Matching()**. VFA determines whether the optional-factor authentication is required after the mandatory-factor authentication, according to the return value of the function **Model-Matching()**.

Algorithm 1 calculates a probability value, which means the degree of being an outlier, through function **PointMatch()**. The function **PointMatch()** compares a pending data point to data points in the corresponding two-dimensional data set. It returns the local outlier probability of the pending data point. The final decision is made according to the sum of the products of the six probability values and the six weight coefficients (line 7 in Algorithm 1) -- when the sum is larger than the threshold τ, the user is considered to be suspicious, otherwise the user is deemed to be a legitimate user.

Algorithm 1 Model-Matching

Input:
SetOfPoints: pending data points extracted from the current login request
Model: the user model,
SetOfWeights: weight coefficients,
Threshold τ: the probability threshold.
Output: "Suspicious User" or "Legitimate User"

```
1: Probability := 0;
2: for i = 1 to 6 do
3:    Point := SetOfPoints.get(i);
4:    Cell := getCell(Point, Model); // returns the cell into which the pending point is placed
5:    ProbabilityP := PointMatch(Point, Cell, Model);
6:    Weight := SetOfWeights.get(i);
7:    Probability += ProbabilityP × Weight;
8: end for
9: if Probability ⩾ τ then
10:    Return "Suspicious User";
11: else
12:    Return "Legitimate User";
13: end if
```

The pseudo code description of the function **PointMatch** is presented as Algorithm 2.

Algorithm 2 PointMatch

Input:
Point: a pending data point,
Cell: the corresponding two-dimensional data set.
Model: the user model.
Output: the Γ value

```
1: SetOfCells := getNeighbor(Cell, Model);
2: while SetOfCells.size < k do
3:    // k is the number of neighbor cells
4:    for i = 1 to SetOfCells.size do
5:       currentCell := SetOfCells.get(i);
6:       currentSet := getNeighbor(currentCell, Model);
7:       SetOfCells.append(currentSet);
8:    end for
9: end while
10: S := findK-Neighbor(SetOfCells, Point);
11: o := Point;
12: Return Γ_S(o);
```

In Algorithm 2, **getNeighbor()** finds the neighbor cells of a specified cell according to Definition 1. The **append()** method of **SetOfCells** inserts cells into a set of cells without repetition. The function **findK-Neighbor()** returns a subset S of data points, which have the following property: the distances of **Point** to them are not larger than the distance between **Point** to its k-th nearest neighbor point [9]. At last, Algorithm 2 computers the local outlier probability Γ of the pending data point based on equation (9).

If the **Model-Matching()** marks the requesting user with suspiciousness, VFA will enable optional-factor authentication. In summary, VFA maximizes the number of authentication factors for a suspicious user but minimizes the number for a legitimate user.

V. Evaluation

In this section, we evaluate VFA from a security perspective and a performance perspective, respectively.

A. Security Analysis

The adversary model is present in Section III, we explain how VFA secures the mobile applications against the adversaries with different levels of power.

1) Against level 1 adversary

The power of the level 1 adversary is the weakest of all. The level 1 adversary does not possess any identity credentials of the victim, including the credentials of the mandatory-factor authentication. By contrast, VFA enables the mandatory-factor authentication for every login request. Therefore, the level 1 adversary cannot pass the mandatory-factor authentication. Namely, VFA can protect the mobile applications against the level 1 adversary.

2) Against level 2 adversary

Comparing with the level 1 adversary, the level 2 adversary possesses the identity credentials of the mandatory-factor authentication, either through phishing or by compromising vulnerable services. The abilities of the level 2 adversary allow him or her to pass the mandatory-factor authentication. But the behaviors of the level 2 adversary do not match with the reference model, because he or she knows nothing about the victim's usage habits. Therefore, VFA will mark the level 2 adversary as a suspicious user, and enable the optional-factor authentication subsequently. In another word, all of authentication factors will be enabled for the level 2 adversary. In this case, the number of authentication factors provided by VFA is the same as MFA. Hence, for the level 2 adversary, VFA can provide security assurances comparable to that of a normal MFA scheme. Considering that the level 2 adversary does not possess the identity credentials of the optional-factor authentication, VFA can secure the mobile applications against the level 2 adversary.

3) Against level 3 adversary

The level 3 adversary has the strongest abilities. Because the level 3 adversary possesses the identity credentials, the mandatory-factor authentication is not a barrier to him or her. The level 3 adversary knows the details of the victim's usage habits, so he or she can imitate the behaviors of the victim to use the mobile applications. This type of abilities can trick VFA into skipping the optional-factor authentication. Therefore, VFA cannot provide security protection when facing the level 3 adversary.

B. Performance

We implemented a prototype to evaluate the performance of VFA. The VFA prototype is aimed at protecting the Weixin client. It consists of a VFA client and a VFA server. The VFA server is responsible for identity management and user model building/matching. The VFA client provides interface to mobile users, and collects signals from mobile devices. When a user tries to launch the VFA prototype, the VFA client enables the mandatory-factor authentication. During the process of the mandatory-factor authentication, both sensor signals and the user's login data are collected by the VFA client, and sent to the VFA server. The VFA server computers the matching degree between the user's behaviors and the user model. If the matching degree was low, namely the user is considered as a suspicious user, the optional-factor authentication is enabled subsequently.

As shown in Fig. 13, the prototype includes 2 authentication factors: a password-based authentication factor and a graph-based authentication factor. The password-based authentication is set as the mandatory authentication factor, whereas the graph-based authentication is set as the optional authentication factor. For a legitimate user, only the password-based authentication is enabled. For a suspicious user, both the password-based authentication and the graph-based authentication are enabled. The total number of lines of code is about 6029, which includes 3006 lines of C code and 3023 lines of Java code.

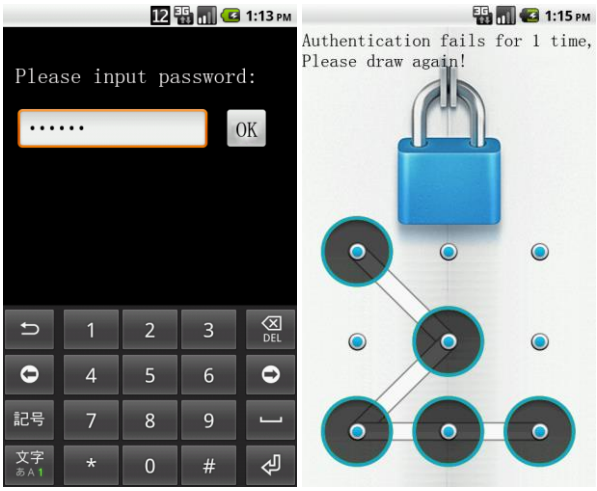

Fig. 13. Authentication factors.

The detailed process of the prototype is described as follows:

a) In the model building stage, the VFA client collects sensor signals and the user's login data from the legitimate user, and sends them to the VFA server.
b) The VFA server builds the user model according to the training data. Once the user model building is done, the number of authentication factors will be dynamically adjusted for a requesting user.
c) The VFA client firstly enables the mandatory-factor authentication when the mobile user tries to launch the VFA prototype. The requesting user needs to provide his password (i.e., the identity credential of the mandatory-factor authentication) to the VFA client.
d) After the VFA client receives the user's password, it sends multiple sensor signals together with the password to the VFA server.
e) By verifying the user's password, the VFA server determines whether or not the requesting user passes the mandatory-factor authentication. If the password is not correct, the server informs the

client to restart the process of the mandatory-factor authentication.

f) After the mandatory-factor authentication, the VFA server determines whether the optional-factor authentication is required, according to the matching degree between the user's behaviors and the user model.
g) If necessary, the requesting user should go through the optional-factor authentication. The VFA client sends the user's graphic password (i.e., the identity credential of the optional-factor authentication) to the VFA server.
h) The VFA server returns the verification result to the VFA client.
i) If the user passes the authentication, the VFA client starts the Weixin client. Otherwise, the VFA client restarts the process of the optional-factor authentication.

TABLE II: A PART OF EXPERIMENTAL RESULTS (THRESHOLD = 10%)

Participants	# of login request	# of only mandatory	# of both mandatory & optional
User 1	36 /week	21 /week	15 /week
User 2	92 /week	49 /week	43 /week
User 3	24 /week	9 /week	15 /week
User 4	22 /week	11 /week	11 /week
User 5	21 /week	8 /week	13 /week
User 6	58 /week	32 /week	26 /week
User 7	29 /week	16 /week	13 /week
User 8	26 /week	19 /week	7 /week
User 9	26 /week	18 /week	8 /week
User 10	38 /week	22 /week	16 /week
…	…	…	…

We conducted an experiment to evaluate the performance of our prototype, which simulated the case of a legitimate user operating the prototype. In the experiment, we collected traces from 21 mobile users (11 males and 10 females), who had participated in our survey described in Section III. We asked all participants to use our prototype for longer than four weeks. This experiment consisted of two parts. The first part focused on collecting data for building user models. It lasted for two weeks. Our prototype used the collected data to generate the user models, which was used for model matching in the second part. The second part of this experiment aimed to evaluating the performance. During this part, our prototype dynamically adjusted the number of authentication factors when the participants attempted to gain access to the VFA prototype. At the same time, the prototype recorded how many times all of the authentication factors were enabled, and how many times only the mandatory-factor authentication was enabled. The second part lasted for two weeks. In the first week of the second part, the threshold value τ was set to 10%, whereas in the second week it was set to 20%. During the second part, the six weight coefficients W_i, W_l, W_a, W_m, W_o and W_e were set to 0.3, 0.3, 0.1, 0.1, 0.1 and 0.1, respectively. Note that an appropriate balance between usability and security can be managed by the administrator, through changing these parameters.

TABLE III: STATISTICAL ANALYSIS FOR EXPERIMENTAL RESULTS

Participants	convenience saving (Threshold = 10%)	convenience saving (Threshold = 20%)
User 1	58.3 %	94.4 %
User 2	53.3 %	68.5 %
User 3	37.5 %	91.7 %
User 4	50.0 %	86.3 %
User 5	38.1 %	66.7 %
User 6	55.2 %	77.6 %
User 7	55.2 %	68.7 %
User 8	73.1 %	84.6 %
User 9	69.2 %	84.3 %
User 10	57.9 %	70.5 %
User 11	48.8 %	63.4 %
User 12	39.7 %	65.5 %
User 13	57.3 %	80.2 %
User 14	51.9 %	72.7 %
User 15	55.1 %	66.1 %
User 16	60.2 %	90.3 %
User 17	50.4 %	71.4 %
User 18	39.3 %	65.5 %
User 19	68.1 %	88.5 %
User 20	35.5 %	53.3 %
User 21	57.2 %	74.3 %

As illustrated in Fig. 14, on average 50.3% of the legitimate login requests only need to go through the mandatory-factor authentication, when the threshold value was set to 10%. As shown in Fig. 15, on average 75.5% of the legitimate login requests are optimized by VFA, when the threshold value was set to 20%. This experiment shows that VFA can reduce the number of authentication factors effectively for a legitimate mobile user.

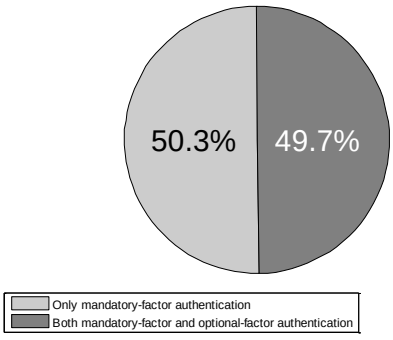

Fig. 14. Performance evaluation ($\tau = 10\%$).

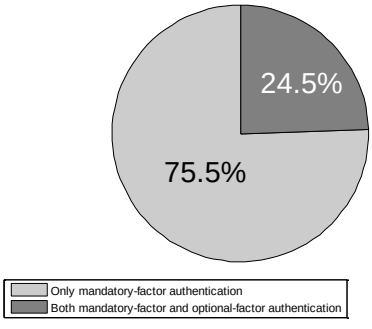

Fig. 15. Performance evaluation ($\tau = 20\%$).

We also conducted another experiment to simulate the case of a stranger attacking the victim's device. This experiment was conducted using the user models built in the first two weeks, without changing any system parameter. In this experiment, we wanted to simulate attacking from a stranger (e.g. when mobile devices was possessed by a stranger). We did this by replacing the user models of Users 1, 3, 7, 8 and 16 with the user model of User 9, and let Users 1, 3, 7, 8 and 16 continue to try to login to Weixin. In this case, the VFA prototype used User 9's model to match rather than the correct models. This experiment lasted for one week.

TABLE IV: EXPERIMENTAL RESULTS WHEN ATTACKING FROM A STRANGER (THRESHOLD = 10%)

Participants	# of login request	# of only mandatory	# of both mandatory & optional
User 1	54 /week	0 /week	54 /week
User 3	19 /week	0 /week	19 /week
User 7	41 /week	0 /week	41 /week
User 8	22 /week	0 /week	22 /week
User 16	35 /week	0 /week	35 /week

As shown in Table IV, all login requests were correctly considered as the requests from an attacker. So both mandatory-factor authentication and optional-factor authentication were enabled for every login request. This experiment shows that our program provides the same security assurance, compared to a normal multi-factor authentication scheme for suspicious users.

VI. CONCLUSIONS

The state-of-the-art approaches for multi-factor authentication set a barrier to usability for mobile users. We propose a generic framework -- VFA -- to lower this barrier through adjusting the number of authentication factors dynamically. The benefit of our approach is that VFA minimizes the number of authentication factors for a legitimate user while providing security assurances comparable to that of conventional multi-factor authentication schemes. However, some more efforts are still needed to deal with the problem of user model updating. In addition, it would be worth some research efforts to enhance the performance of abnormal detection.

APPENDIX A QUESTIONNAIRE

Fig. 16 is the first page of the questionnaire for the user study. The questions touch on when, where and how the mobile users use the special applications. We perform a statistical analysis to get the result.

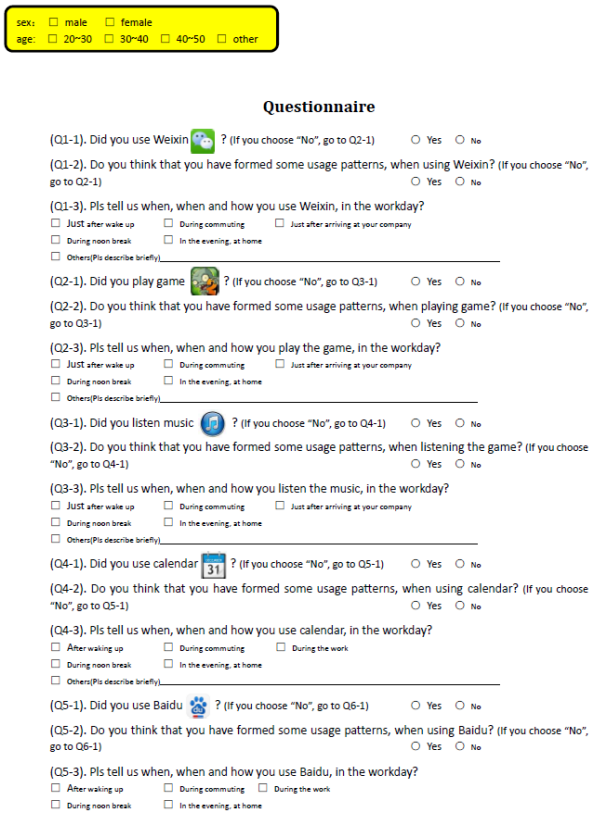

Fig. 16. Questionnaire for user study.

ACKNOWLEDGMENT

We would like to thank our team members and all anonymous reviewers for helping us make this paper better.

REFERENCES

[1] A. Singer, "Tempting fate," *Login*, vol. 30, no. 1, pp. 27-30, 2005.
[2] L. Nixon, "The stakkato intrusions: What happened and what have we learned?" in *Proc. 6th IEEE International Symposium on*

Cluster Computing and the Grid Workshops, Singapore, May 2006, pp. 27.
[3] V. Hazlewood, P. Kovatch, M. Ezell, M. Johnson, and P. Redd, "Improved grid security posture through multi-factor authentication," in *Proc. 12th IEEE/ACM International Conference on Grid Computing*, Lyon, France, September 2011, pp. 106–113.
[4] G. Yang, D. Wonga, H. Wang, and X. Deng, "Two-factor mutual authentication based on smart cards and passwords," *Journal of Computer and System Sciences*, vol. 74, no. 7, pp. 1160-1172, 2008
[5] A. T. B. Jin, D. N. C. Ling, and A. Gohb, "Biohashing: Two factor authentication featuring fingerprint data and tokenised random number," *Pattern Recognition*, vol. 37, no. 11, pp. 2245 - 2255, 2004.
[6] A. Lumini and L. Nanni, "An improved BioHashing for human authentication," *Pattern Recognition*, vol. 40, no. 3, pp. 1057 - 1065, 2007.
[7] T. Connie, A. Teoh, M. Goh, and D. Ngo, "PalmHashing: A novel approach for dual-factor authentication," *Pattern Analysis and Applications*, vol. 7, no. 3, pp. 255 - 268, 2004.
[8] X. Huang, Y. Xiang, J. Zhou, and R. H. Deng, "A generic framework for three-factor authentication: Preserving security and privacy in distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1390-1397, 2011.
[9] H. P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "LoOP: Local outlier probabilities," in *Proc. 18th ACM conference on Information and knowledge management*, Hong Kong, China, November 2009, pp. 1649-1652
[10] R. Greenstadt and J. Beal, "Cognitive security for personal devices," in *Proc. 1st ACM workshop on Workshop on AISec*, Alexandria, Virginia, USA, October 2008, pp. 27-30.
[11] E. Shi, Y. Niu, M. Jakobsson, and R. Chow, "Implicit authentication through learning user behavior," in *Proc. 13th International Conference on Information Security*, Boca Raton, FL, USA, October 2010, pp. 99–113.
[12] M. Jakobsson, E. Shi, P. Golle, and R. Chow, "Implicit authentication for mobile devices," in *Proc. 4th USENIX Workshop on Hot Topics in Security*, Montreal, Canada, August 2009.
[13] Y. Niu, E. Shi, and R. Chow, "One Experience Collecting Sensitive Mobile Data," in *Proc. Usable Security Experiment Reports (USER) Workshop of SOUPS*, Redmond, WA, USA, July 2010.
[14] R. Chow, M. Jakobsson, R. Masuoka, J. Molina, Y. Niu, E. Shi, and Z. Song, "Authentication in the clouds: A framework and its application to mobile users," in *Proc. ACM Workshop on Cloud Computing Security Workshop*, Chicago, Illinois, USA, October 2010, pp. 1–6.
[15] A. Czeskis, M. Dietz, T. Kohno, D. Wallach, and D. Balfanz, "Strengthening user authentication through opportunistic cryptographic identity assertions," in *Proc. 19th ACM Conference on Computer and Communications Security*, Raleigh, NC, USA, October 2012, pp. 404–414.
[16] O. Riva, C. Qin, K. Strauss, and D. Lymberopoulos, "Progressive authentication: Deciding when to authenticate on mobile phones," in *Proc. 21st USENIX Conference on Security Symposium*, Bellevue, WA, USA, August 2012, pp. 301–316.
[17] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, Oregon, USA, August 1996, pp. 226–231.
[18] Y. Zhao and J. Song, "AGRID: An efficient algorithm for clustering large high-dimensional datasets," in *Proc. 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Seoul, Korea, April 2003, pp. 271 - 282
[19] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD International Conference on Management of Data*, Dallas, Texas, USA, June 2000, pp. 93–104.
[20] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Information Systems*, vol. 32, no. 7, pp. 978-986, 2007.
[21] H. S. Kim, S. Gao, Y. Xia, G. B. Kim, and H. Y. Bae, "DGCL: An efficient density and grid based clustering algorithm for large spatial database," in *Proc. 7th International Conference on Web-Age Information Management*, Hong Kong, China, June 2006, pp. 362–371.
[22] D. Yang, E. A. Rundensteiner, and M. O. Ward, "Summarization and matching of density-based clusters in streaming environments," in *Proc. 38th International Conference on Very Large Databases*, Istanbul, Turkey, August 2012, pp. 121-132.
[23] Y. Zhao and J. Song, "GDILC: A grid-based density-isoline clustering algorithm," in *Proc. International Conferences on Info-tech and Info-net*, Beijing, China, October 2001, pp. 140–145.
[24] E. Ma and T. Chow, "A new shifting grid clustering algorithm," *Pattern Recognition*, vol. 37, no. 3, pp. 503-514, 2004.
[25] Y. Ma, H. Shi, H. Ma, and H. Wang, "Dynamic process monitoring using adaptive local outlier factor," *Chemometrics and Intelligent Laboratory Systems*, vol. 127, pp. 89-101, 2013.
[26] J. Liu and H. Deng, "Outlier detection on uncertain data based on local information," *Knowledge-Based Systems*, vol. 51, pp. 60-71, 2013.

Kai Chen was born in Tianjin Province, China, in 1985. He received the B.S. degree from the Tianjin University of Science and Technology (TUST), Tianjin, in 2007 and the M.S. degree from the University of Chinese Academy of Science (UCAS), Beijing, in 2012. He is current pursuing the Ph.D. degree in information security, UCAS. His research interests include network security, authentication and industrial control system security.

Weifeng Chen obtained his Ph.D. in Computer Science from the University of Massachusetts at Amherst in 2006, and his M.S. and B.S. in computer science from Chinese Academy of Science and Beijing University in 2001 and 1998, respectively. He is currently a tenured Associate Professor in the Department of Math and Computer Science at California University of Pennsylvania (CalU). He is also the director of the PSM Cybersecurity program at CalU. He was an Assistant Professor at CalU 2007-2012. From 2006-2007, he was an Assistant Professor at the City University of New York. His research interests focus on computer network, security and privacy.

Dongdai Lin received the M.S. and Ph.D. degrees in fundamental mathematics from the Institute of Systems Science, Chinese Academy of Sciences, Beijing, China, in 1987 and 1990, respectively. He is currently the Director of the National Engineering Research Center for Information Security and Deputy Director of the State Key Laboratory of Information Security, Institute of Information Engineering of Chinese Academy of Sciences. He has published more than 150 research papers in journals and conference proceedings. His current research interests include cryptology, security protocols, symbolic computation, and software development, and multivariate public key cryptography, sequences and stream cipher, zero knowledge proof, and network-based cryptographic computation.

Zhen Xu received the M.S. and Ph.D. degrees from the Institute of Software Chinese Academy of Sciences, Beijing, China. He is currently the research professor at Institute of Information Engineering Chinese Academy of Sciences. His current research interests include network security, trusted computing and cloud security.

Yazhe Wang received the Ph.D. degrees from the Institute of Software Chinese Academy of Sciences, Beijing, China. He is currently the associate research professor at Institute of Information Engineering Chinese Academy of Sciences. His current research interests include authentication, authorization and smart device security.