YAGP: Yet Another Graphical Password Strategy

Haichang Gao, Xuewu Guo, Xiaoping Chen, Liming Wang, and Xiyang Liu
Software Engineering Institute, Xidian University 710071, China
{hchgao, xyliu}@xidian.edu.cn

Abstract

Alphanumeric passwords are widely used in computer and network authentication to protect users’ privacy. However, it is well known that long, text-based passwords are hard for people to remember, while shorter ones are susceptible to attack. Graphical password is a promising solution to this problem. Draw-A-Secret (DAS) is a typical implementation based on the user drawing on a grid canvas. Currently, too many constraints result in reduction in user experience and prevent its popularity. A novel graphical password strategy Yet Another Graphical Password (YAGP) inspired by DAS is proposed in this paper. The proposal has the advantages of free drawing positions, strong shoulder surfing resistance and large password space. Experiments illustrate the effectiveness of YAGP.

1. Introduction

Conventionally, alphanumeric text passwords are used for user authentication. However, it is challenging for users to remember long complicated passwords, while short simple passwords are too weak to resist attackers and are therefore a security risk. Such disadvantages of alphanumeric password seem to put users in dilemma.

Psychologists have shown that with both recognition and recall tasks, images are more memorable than words or sentences [1]. This is encouraging in terms of memory for graphical passwords [3]. The existing graphical password techniques can be divided into three general categories: recognizing the pass-images, repeating a sequence of actions and reproducing a drawing. Draw-A-Secret (DAS) is a representative “reproducing a drawing” scheme which is of particular interest and worthy of extensive study [2]. The property of alphabet independence liberates users from remembering any text password and makes the scheme accessible to speakers of any language.

Background Draw-A-Secret (BDAS) provides a significant extension to the security and usability of the DAS scheme by introducing background images [4]. The background images can encourage users to set strong passwords and thus enhance memorability. Qualitative Draw-A-Secret (QDAS) also extends the ideas pioneered within DAS [5]. The use of qualitative spatial relations relaxes the tight constraints on the reconstruction of a secret, allowing a range of deviations from the original. In DAS based schemes, the user password is a free-form drawing produced on an N×N grid, typically a 5×5 grid. There are many restrictions on drawing such as keeping every stroke off the grid lines and ensuring redrawing in the exact position, which make it difficult for the user himself to recall the secret.

A novel graphical password scheme, Yet Another Graphical Password (YAGP), is proposed in this paper. YAGP inherits the strongpoint of DAS and relaxes the restrictions for users. First, the exact stroke positions are no longer required, encouraging greater user concentration on the image. Second, YAGP provides a trend-sensitive judgment mechanism when authenticating the reentered passwords. It also possesses a personality-related capability of resisting shoulder surfing using individual drawing style. Finally, a larger password space is obtained using more precise grid granularity.

The remainder of the paper is outlined as follows: Section 2 briefly reviews the current state of the art of graphical passwords, but mainly discusses DAS and its shortcomings. Section 3 defines the password format, describes the comparison algorithm, introduces similarity computation, and discusses security issues. In Section 4, some preliminary experiments and results are demonstrated. Finally, Section 5 concludes.

2. Related Works

Nowadays, alphanumeric passwords are mostly used for computer and network authentication. Most people tend to choose simple text passwords [6]. However, such passwords are not strong enough to
resist attackers. The saying, “a picture is worth a thousand words”, indicates that pictures usually contain much more information, and at the same time, psychological research has shown images are more memorable than words or sentences. Therefore, the study of graphical passwords is increasing and draws much attention from many researchers.

The existing graphical password techniques can be divided into three general categories: pass-images recognition (Deja vu[7], Passfaces[8], Convex hull[9] etc.), action sequence repetition (V-GO[10], Passpoints[3] etc.) and drawing reproduction (DAS[2], Syukri[11]).

![Figure 1. The generation of DAS password. (a) shows the user’s drawing on the grid canvas. (b) is the internal representation of the user’s secret. (c) depicts the storage of the derived key.](image)

DAS is a representative “drawing reproduction” scheme in which the password is a simple picture drawn on a grid. It is a purely graphical password scheme based on recall, and requires the user to create a free-form image on a drawing grid. The drawing is mapped to a sequence of coordinate pairs by listing the cells through which the drawing passes in the order in which it passes through them, with a distinguishing coordinate pair inserted in the sequence for each “pen up” event, i.e., whenever the user lifts the stylus from the drawing surface. Fig.1 shows how DAS works. DAS constitutes a much larger password space than text passwords [2].

![Figure 2. Three example symbols.](image)

If we draw a secret within one cell of the grid, there would be no difference between the internal representations of a complicated symbol, a checkmark and a dot. As illustrated in Fig.2, the three symbols have the same internal representation if their coordinates are not taken into account.

In addition, DAS imposes strict rules on the location of each stroke drawn by the user, posing a major challenge for most users. Users cannot locate strokes too close to a grid-line or cross a corner, as DAS does not accept such a secret. However, it is very difficult for users to avoid violating these rules as illustrated in Fig.3. To replicate a password, the user must cross the same cells, in the same order, lift the pen in the same place, and make no corner crossings [4].

![Figure 3. Examples of rule violations in DAS. (a) Lines near grid line. (b) Endpoints near grid line. (c) Strokes near cell corner.](image)

Recently, some new schemes based on DAS have emerged. BDAS uses background pictures to induce the user to draw a more complicated secret [4]. Meanwhile, the background image helps the user to recall the secret more efficiently and precisely. However, the experimental results reveal that the performance of BDAS has close relationship to users’ artistic skills [4]. To take best advantage of BDAS users should have training first, an obstacle to the popularity of that scheme. QDAS extends DAS by introducing grid transformation to enhance the precision of graphical password [5]. It allows users to set strong secrets that do not impose a strain on long-term memory and are resistant to shoulder surfing. The use of qualitative spatial relations relaxes the tight constraints of DAS, but it still does not solve the problems completely.

3. Design and Implementation of YAGP

In this section, a novel position-free graphical password (YAGP) is proposed. YAGP inherits the advantages of other graphical password schemes, such as DAS. Moreover, it has its unique characteristics including permitting redrawing anywhere on the grid canvas, and analyzing user drawing style to resist shoulder surfing.

3.1. Password Format
In the grid-based interface, every stroke of an image consists of three types of elements: pen-down, pen-moving and pen-up. To denote a stroke effectively and efficiently, the concept of neighbor in DAS is extended and a coding rule is introduced.

Let \( N(x, y) \) be the eight neighbor positions of a cell \((x, y)\), it can be represented as follows:

\[
N(x, y) = \{ (x-1, y-1), (x, y-1), (x+1, y-1), (x-1, y), (x+1, y), (x-1, y+1), (x, y+1), (x+1, y+1) \}.
\]

As Fig.4 shows, the numbers 1, 2, 3, 4, 6, 7, 8, 9 are used to represent position information of the current cell along with pen moving as a matter of convenience. This is also the coding rule of graphical password in YAGP. So in this way, each pen-moving obtains a code. Pen-up and pen-down in a stroke are coded by ‘5’. Thus, each stroke can be denoted by a coded string bounded at both ends by a ‘5’.

An example “umbrella” drawing is depicted in Fig.5, which consists of two strokes. The code is generated as follows: the first pen-down makes a ‘5’, and the moving of the stylus produces the consecutive coded string ‘7777766666666611111111’. Finally, the pen-up is also marked as a ‘5’. Therefore, the coded string for the first stroke is ‘5777776666666666666111111111’. The second stroke generates coded string in the same way. As a result, the coded string of the whole drawing is ‘577777666666666666611111115555555’. 

3.2. Description of the Comparison Algorithm

By comparing the reentered graphical password with the predefined one, there are three types of matching.

1. Complete matching, i.e., the reentered graphical password matches the predefined completely. That is to say, both of the graphical passwords have the same number of strokes and each two corresponding stroke codes are identical. In such a case, the authentication is completed successfully.

2. Mismatching, which means the reentered graphical password does not have the same number of strokes as the predefined one, or the Trend Quadrants sequences of the former and the latter are not the same, i.e., the numbers of divided substrings are different. Such cases can be treated as a mismatch.

3. Partial matching, which means there is a similarity value between the reentered graphical password and the original. This is the most common case. Such a match can be considered successful if the value is higher than a predefined threshold value. Otherwise, it becomes a mismatch.

Three stages of comparison are required to compare the reentered graphical password with the predefined one. First, the stroke numbers of the original image and the reentered one should be matched. Second, the Trend Quadrants sequences of the stroke pairs should be the same. Finally, the similarity of the two images should be measured. If the similarity value is larger than a predefined threshold, the authentication can be considered to be successful. Otherwise, it will be determined as a failure.

Let \( N \) be the number of strokes in the original graphical password, \( N’ \) be the number of strokes in the reentered password, and \( S \) be the similarity of two coded strings. Based on these definitions, the comparison algorithm is described as Fig.6.

<table>
<thead>
<tr>
<th>Algorithm 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 Initialization, ( S=0 );</td>
</tr>
<tr>
<td>Step 2 Count ( N ) and ( N’ );</td>
</tr>
<tr>
<td>[ \text{If } N\neq N’, \text{ goto Step 4; Otherwise, continue}; ]</td>
</tr>
<tr>
<td>Step 3 For ( i=1 ) to ( N ) Do</td>
</tr>
<tr>
<td>[ \text{Compare the trends of the } i\text{th stroke pair ;} ]</td>
</tr>
<tr>
<td>[ \text{If they don’t share the same trend, goto Step4; Otherwise, continue}; ]</td>
</tr>
<tr>
<td>End For;</td>
</tr>
<tr>
<td>[ \text{Compute } S; ]</td>
</tr>
<tr>
<td>Step 4 Return ( S ).</td>
</tr>
</tbody>
</table>

Figure 6. Algorithm 1.
Trend Quadrants, and the divisions of a stroke which assist the similarity computation are introduced.

Using the umbrella graphic as an example, Fig.5 represents the original graphical password image which the user predefines, and Fig.7 represents the reentered image to be authenticated. The corresponding coded strings can be depicted as "5777776666666666666111 1155888888771125" and "57777766366111 155888888877441225" respectively, using the representing strategy described above.

Intuitively, the two graphics look alike on the whole, but there are some minor differences. The current task is to determine the similarity between the two coded strings, i.e., the similarity between images in Fig.5 and Fig.7, so as to decide on the legality of the authentication according to the predefined threshold value.

To compute the similarity and obtain a more effective comparison, some concepts regarding Levenshtein distance and other original methods are introduced first.

### 3.3.1. Levenshtein Distance

There are a number of algorithms used to measure the similarity between two approximate strings, with the most used being the Levenshtein distance \([12]\). It is also adopted in our YAGP strategy.

The Levenshtein distance between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation can be an insertion, deletion, or substitution of a single character. It is often implemented by constructing a distance matrix. For example, the Levenshtein distance between "micky" and "monkey" is 3, since the following three operations change one into the other, and there is no method to do it with fewer operations:

1. micky → mocky (substitution of 'o' for 'y');
2. mocky → monkey (substitution of 'n' for 'c');
3. monky → monkey (insert 'e' between 'k' and 'y').

Let \(str1 = p_1p_2⋯p_m\) and \(str2 = t_1t_2⋯t_n\), where \(m\) and \(n\) be the length of each string. Let \(D\) be an \(m+1\) by \(n+1\) matrix where \(D(i, j)\) be the Levenshtein distance between \(p_i, p_2⋯p_m\) and \(t_1t_2⋯t_j\). Let \(LD(str1, str2)\) be the Levenshtein distance between \(str1\) and \(str2\) which equals \(D(m, n)\) located in the lower right hand corner of the matrix.

Herein,

\[
\begin{align*}
D(0, j) &= j, \quad 0 \leq j \leq n \\
D(i-1, j+1) &= 1 \\
D(i, j) &= \min \left[ D(i-1, j-1)+1 \text{ if } p_i=t_j \text{ then } 0 \text{ else } 1 \right]
\end{align*}
\]

The distance matrix constructed for the above example can be represented in Fig.8, and the Levenshtein distance is the bottom-right element ‘3’ of the matrix.

### 3.3.2. Trend Quadrants

In order to compare the stroke trends between two drawings, a concept named Trend Quadrant is proposed here, illustrated in Fig.9. Trend Quadrant I denotes an up-right trend. In the same way, Trend Quadrant II, III and IV denote up-left, down-left and down-right trends respectively.

Let \(X = x_1x_2⋯x_n\), \(n \geq 0\) be the coded string of a stroke, where \(x_i\) be the \(i\)th character of \(X\). The length of the string is \(|X|=n\), where \(n=0\) means that the string represents a point stroke only consisting of a pen-down and a pen-up. The Trend Quadrant of \(i\)th cell relative to its previous cell can be determined definitely if \(x_i \in \{1, 3, 7, 9\}\). To determine the Trend Quadrant if

![Figure 7. The reentered umbrella.](image)

![Figure 8. Distance matrix.](image)

![Figure 9. Trend Quadrants.](image)
x_i \in \{2,4,6,8\}, more information is needed as the single character in such a set cannot make a final decision on an exact Trend Quadrant.

Based on the Trend Quadrant concept described above, the umbrella example in Fig.5 containing two strokes can be divided into five sequenced trends: III→I(IV)→II→III→II as illustrated in Fig.10. The second trend I(IV) means that the corresponding segment is in Trend Quadrant I or IV, but is still uncertain as to which exact Trend Quadrant it belongs to. That is to say, it may have the same trend with a segment which is up-right or down-right.

3.3.3. Division of a Stroke. Usually, the graphical passwords consist of several strokes. Study shows that the shapes of secret images are mostly influenced by user personality. For example, different people may have different styles when drawing a same letter ‘Q’ on the grid canvas as illustrated in Fig.11.

A 4-bit string q_i consisting of ‘0’ and ‘1’ is defined to represent which Trend Quadrant the ith character of the coding string X = x_1x_2...x_n belongs to. Let q_i = b_4b_3b_2b_1 , where b_4, b_3, b_2, b_1 indicate whether x_i is in Trend Quadrant I, II, III, IV respectively, where b_j = 1(1 \leq j \leq 4) means x_i may belong to the corresponding Trend Quadrant.

According to the Trend Quadrants concept, we can obtain Equation (2).

\[
q_i = \begin{cases} 
0010, & x_i = 1 \\
0011, & x_i = 2 \\
0001, & x_i = 3 \\
0110, & x_i = 4 \\
1001, & x_i = 6 \\
0100, & x_i = 7 \\
1100, & x_i = 8 \\
1000, & x_i = 9 
\end{cases}
\]

Let Q_i = Q_i−1 & q_i be the ADD result of q_i and its foregoing total ADD result. The stroke trend is holding if and only if Q_i ≠ 0000. Once Q_i = 0000 is obtained, it implies that a different trend is encountered, where we divide the stroke into separate trend segments.

The reason for dividing a stroke into several segments is to compare number of the trends and evaluate the similarity of corresponding segments of the original and reentered graphical passwords. Only if the two counts are equal and the similarity is higher than a predefined threshold value, is the authentication proved to be a legal one.

We denote by F_s = x_{k1}…x_{ki}, 1 \leq k < i \leq n a substring of X with a trend. Given a coded string within a stroke, the division steps are described as algorithm 2 in Fig.12.

**Algorithm 2**

Step 1 Initialization of the variables
(k = 1, s = 1, i = 1, Q_0 = 1111);

Step 2 While i < (n + 1) Do

- Compute q_i using Equation (2);
- Q_i = Q_{i−1} & q_i;

- If Q_i = 0000 Do 
  F_s = x_{k1}…x_{ki}, k = i, s++ Q_s = q_i;

- i++;

- End While;

F_s = x_{k1}…x_{kn};

Step 3 X = F_1…F_s;

**Figure 12. Algorithm 2.**

The divided substrings of X are F_1, F_2, …, F_s, and the number of the substrings is s. Table 1 shows the division of coded strings “57777776666666666611 1155888888771125” and “57777776666666666611 155888888877441225” generated from Fig.5 and Fig. 7. Their corresponding divided images are depicted by
different colors as shown in Fig.13. The division results correspond perfectly to the Trend Quadrants sequence described in Section 3.3.2.

### Table 1. Division of coding strings of umbrella images in Fig. 13.

<table>
<thead>
<tr>
<th>Substring (a)</th>
<th>Substring (b)</th>
<th>Trend Quadrant</th>
</tr>
</thead>
<tbody>
<tr>
<td>F₁</td>
<td>777777</td>
<td>III</td>
</tr>
<tr>
<td>F₂</td>
<td>666666666666</td>
<td>IV</td>
</tr>
<tr>
<td>F₃</td>
<td>11111</td>
<td>II</td>
</tr>
<tr>
<td>F₄</td>
<td>888888777</td>
<td>III</td>
</tr>
<tr>
<td>F₅</td>
<td>112</td>
<td>II</td>
</tr>
</tbody>
</table>

![Figure 13. The two divided images. (a) The divided original images. (b) The divided reentered images.](image)

### 3.3.4. Similarity Algorithm.

Before the similarity comparison of two coded strings is carried out, two steps of authentication must be passed. That is, (1) the two corresponding images should share the same number of strokes, (2) and each corresponding stroke in the two images should have the same Trend Quadrants sequence.

The coded strings of two images can be denoted by $X = F₁ \cdots F_j \cdots F_m$ and $X' = F'₁ \cdots F'j \cdots F'_m$, where $F_j$ and $F'_j$ are the $j$th substrings separately, and $M$ is the total substring number of each coded string. The difference between $F_j$ and $F'_j$ can be defined as:

$$d(j) = \frac{\sum_{i=0}^{M} \| F_j[i] \| - \| F'_j[i] \|}{\max(\| F_j \|, \| F'_j \|)}$$  \hspace{0.5cm} (3)

Let $X_i$ be the coded string of the $i$th stroke in the original graphical password, and $N$ be the stroke number of either images. Then the similarity $S$ of two images can be defined as:

$$S = 1 - \frac{\sum_{i=0}^{N} d(j)}{\sum_{i=0}^{N} \| X_i \|}$$  \hspace{0.5cm} (4)

Algorithm 3 in Fig.14 is the complete algorithm that performs the three comparison steps. It provides a detailed description of Algorithm 1.

### Algorithm 3

Step 1 Initialization, $S=0$, $M=0$, $D=0$, $L=0$;
Step 2 Compute $N$ and $N'$;
If $N \neq N'$, go to Step 5;
Otherwise, $X = X₁ \cdots X_j \cdots X_N$

$$X' = X'_₁ \cdots X'_j \cdots X'_N$$

Step 3 For $i=1$ to $N$ Do
Divide the two strings of the $i$th stroke using Algorithm 2, $X_j = F_j₁ F_j₂ \cdots F_jₖ$

If $k \neq l$, go to Step 5; Otherwise, $M = k$

$L⁺ = \| X_j \|$;
End For;
Step 4 For $i=1$ to $M$ Do
Compute the difference $d(i)$ using Equation (3);

$$D⁺ = d(i)$$

End For;

$$S = 1 - \frac{D}{L}$$

Step 5 Return $S$.

According to Algorithm 3, the similarity between (13.a) and (13.b) in Fig.13 can be proceeding as follows. $d(1) = 0$, $d(2) = 1$, $d(3) = 1$, $d(4) = 2.18$, $d(5) = 1$, $D = \sum_{i=0}^{5} d(i) = 5.18$, $L = \sum_{i=0}^{5} \| X_i \| = 34$, $S = 1 - \frac{5.18}{34} = 0.8476$. Under the moderate threshold value (for example, 80%), the authentication is successful.

### 3.4. Security

To a significant degree, security is influenced by password space size. In most existing DAS-based graphical password schemes, the password space size is determined mainly by the grid density, the stroke number and the length of each stroke. YAGP adopts a grid of $48 \times 64$ denser than the $5 \times 5$ grid used in DAS, as demonstrated by the experiment results in Section 4.1. With such a dense grid, it becomes easier for users to draw long strokes. Furthermore, YAGP imposes no limitation to the number of strokes. Therefore, YAGP
can provide a larger password space than most existing graphical password schemes.

YAGP can also prevent the brute force attacks effectively. The experiment results show that users tend to set a graphical password at the average length of 100. If the attacker wants to break the correct password by a brute force attack, in the worst case it may take approximately $8^{100}$ times for successful authentication, which is an impractical process.

YAGP obtains a good performance in resisting shoulder surfing. First, YAGP is a position-free scheme, the user can draw his graphical password anywhere on the canvas, which makes shoulder surfing a difficult task. For example, the user can make a drawing in a small corner where it is harder to peep. Second, the stroke sequence cannot be reflected by the graph in YAGP, and authentication process sees it as a critical checking factor. This property ensures that the peeper still cannot sign in even if he glimpses the images, because he could not recall the correct stroke sequence set by the legal user. Finally, YAGP takes into account the drawing trends, which means it records the user drawing style to a certain extent. Therefore, the security is greatly enhanced as personality is hard to imitate.

As a consequence, YAGP has credible security both in the password space and resisting shoulder surfing.

The proposed YAGP system is implemented using C++ language and is available at https://sourceforge.net/projects/yagp-xidian/. To ensure the security of user graphical passwords stored in YAGP, DES encryption is adopted in the implementation. The interface has a grid canvas with a granularity of 48×64 as illustrated in Fig.15.

4. Preliminary Experiments

Experiments were performed to evaluate the proposed YAGP strategy. For the study, we targeted a population of experienced computer users. The participants were 30 university members, including 4 teachers and 26 college students. The majority of the students were studying for their Master’s degrees. The average age of the participants was 26 years old. All of the users were familiar with PCs.

Figure 15. The YAGP system Interface (48×64 density grid).

4.1. Grid Granularity Selection

The grid canvas adopted by YAGP is a 3.5 inch canvas widely used in PDA devices with a width-to-length ratio of 3:4. A grid granularity of 5×5 is used in DAS. However, such a rough granularity is not precise enough to express complex graphical passwords. A fine-grained grid is used in YAGP. Experiments were carried out on five groups of granularity (15×20, 30×40, 48×64, 60×80, and 120×160). Results show that a grid of 48×64 is the most suitable choice for security and usability.

The first stage experiments of grid granularity selection lasted 7 days approximately. As an introduction, the 30 participants were given ten minutes to become familiar with the YAGP system. First, each participant was asked to draw graphical passwords in a 15×20 grid canvas, and then redraw the graphical passwords to authenticate. At the same time, every participant peeked at his neighbors’ graphical password and attempted to attack. Both the legal user and the attacker could redraw a maximum of three times, and the greatest similarity of each participant’s drawing to the original image was recorded. The experiments were carried out several times with grid of different density. According to the similarity of each participant’s redrawing, the distribution of participants is calculated and shown in Table2. The total number in the table showed the participants who have a similarity value successfully. Some people failed on the equivalent stroke numbers or substring numbers that must be obtained on register and authentication phases, so they didn’t get a score.

Table 2 shows that under the circumstance of 15×20 grid, only 20 of 30 legal users can redraw their graphical passwords, as such a coarse grained grid cannot represent the graphical password information well. We also found that, in general, fine-grained grids achieve better validation results. But that is not to say, the finer grained the grid, the better the validation. We can see from the table, the validated number under a density of 120×160 grid is lower than that in 48×64. The reason is that too dense a grid makes the drawing trend more changeable, and therefore harder to recall accurately. After numerous experiments, a compromise was achieved with a 48×64 grid.
The average length of the graphical passwords was evaluated based on the results of the foregoing testing. In a 48×64 grid, the analysis of the lengths of all the 30 graphical passwords showed that, the graphical password in YAGP had a general length range from 100 to 200 which contains a huge password space.

From the table, we can also see that the YAGP scheme demonstrated a significantly improved performance in resisting shoulder surfing. The successful shoulder surfing occurred when the user created a very simple graphical password, such as a single line.

Table 2. The distribution of participants according to similarity (Total 30 participants).

<table>
<thead>
<tr>
<th>Density</th>
<th>Participant distribution (legal user number/ attacker user number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15×20</td>
<td>0/0 0/0 0/0 1/1 2/0 4/0 3/0 6/1 2/0 2/0 20/2</td>
</tr>
<tr>
<td>30×40</td>
<td>0/0 0/0 0/1 1/2 1/1 5/0 7/1 9/0 3/0 1/0 27/5</td>
</tr>
<tr>
<td>48×64</td>
<td>0/0 0/1 0/0 0/1 1/1 7/2 9/1 9/0 2/0 2/0 30/6</td>
</tr>
<tr>
<td>60×80</td>
<td>0/0 0/1 1/0 0/1 2/1 9/2 7/1 7/0 2/0 1/0 29/6</td>
</tr>
<tr>
<td>120×160</td>
<td>1/0 1/0 2/1 3/0 7/1 3/2 4/0 3/0 0/0 0/0 24/4</td>
</tr>
</tbody>
</table>

4.2. Threshold Value Selection

The second phase was designed to determine a reasonable threshold value of similarity, substantially affecting the authentication results. If the threshold value is too large, it has a high probability of blocking the legal user. If the threshold value is too small, it cannot effectively resist shoulder surfing.

Table 3. The count of the successful authentications with different threshold values.

<table>
<thead>
<tr>
<th>Density</th>
<th>Legal users</th>
<th>Shoulder surfers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One time</td>
<td>Three times</td>
</tr>
<tr>
<td>55%</td>
<td>18</td>
<td>1</td>
</tr>
<tr>
<td>60%</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>65%</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>70%</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>75%</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>80%</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2 shows that in a 48×64 grid, the similarity of legal users’ drawings mainly focuses on the range from 55% to 80%, while similarity of attackers’ from 40% to 65%. Therefore, in order to keep the system security and not to block the legal users, the threshold value should be certain value between 55% and 80%. Another 18 participants took part in the second stage experiment. Each person was asked to draw his secret six times, and each time with a different threshold value (from 55% to 80%). The one-time pass through validation and three-time pass through validation numbers are recorded in Table 3. At the same time, every participant was requested to peek at his neighbors’ graphical password and attempt to attack. These experiments were conducted over 2 days, and the experimental results are shown in Table 3.

The results illustrated in Table 3 suggest 60% be the most suitable threshold value of similarity comparison in YAGP, since it can validate the most legal users and effectively resist shoulder surfing. In conclusion, a 48×64 density grid and a similarity threshold of 60% are the best choices for our YAGP strategy.

4.3. Memorability

After the threshold value was determined, the third phase experiments were aimed at the memorability of YAGP. Two days after the participants had set their passwords, they were asked to perform the authentication over again, and given a maximum of three times to redraw their passwords. We had found that 27 out of 28 graphical passwords were recalled successfully. Two weeks later, 15 participants redrew their passwords once again and three chances were permitted. The results showed that 13 participants recalled the graphical passwords successfully. The experimental results showed a better memorability performance.

5. Conclusions and Future Work

A novel graphical password scheme is proposed in this paper and some preliminary experiments are carried out. The results show that YAGP achieves an encouraging performance in usability and security and possesses a high resistance to shoulder surfing.

In a 48×64 grid, the secret drawings can be described in detail. The users can concentrate on the
drawing to improve user experience because exact positions are not required in YAGP. Meanwhile, the algorithm proposed in YAGP is trend-sensitive which actually reflects drawing trends. Furthermore, user personalities have a great influence on the drawings and therefore make it harder for others to imitate. Additionally, users can draw the secrets small enough to resist shoulder surfing.

The main drawback of YAGP is that it’s hard to redraw the password precisely. The legal user cannot always be assured to login successfully because the gaps between user drawings are uncertain while the similarity threshold value is fixed.

Future research will concentrate on improving YAGP as well as developing a comparison algorithm of higher efficiency in distinguishing the legal user from attackers.

6. Acknowledgement

The authors would like to thank the anonymous reviewers for their careful reading of this paper and for their helpful and constructive comments. This work was supported in part by the Defence Pre-Research project of China under grant no. 51315050105 and the Defence Foundation of China under grant no. 9140A15050206DZ01.

7. References


