NAVI: Novel Authentication with Visual Information

Emmanouil Georgakakis, Nikos Komninos, Christos Douligeris
Department of Informatics. University of Piraeus
Piraeus, Greece
egeo@unipi.gr, nkom@ieee.org, cdoulig@unipi.gr

Abstract— **Text-based passwords, despite their well-known drawbacks, remain the dominant user authentication scheme implemented. Graphical password systems, based on visual information such as the recognition of photographs and / or pictures, have emerged as a promising alternative to the aggregate reliance on text passwords. Nevertheless, despite the advantages offered they have not been widely used in practice since many open issues need to be resolved. In this paper we propose a novel graphical password scheme, NAVI, where the credentials of the user are his username and a password formulated by drawing a route on a predefined map. We analyze the strength of the password generated by this scheme and present a prototype implementation in order to illustrate the feasibility of our proposal. Finally, we discuss NAVI's security features and compare it with existing graphical password schemes as well as text-based passwords in terms of key security features, such aspassword keyspace, dictionary attacks and guessing attacks. The proposed scheme appears to have the same or better performance in the majority of the security features examined.**

Keywords-component; graphical passwords; knowledge-based authentication;

I. INTRODUCTION

In challenge response authentication schemes the authenticator poses a question – a challenge- and the user has to provide a correct answer – a response- in order to be authenticated. The simplest challenge-response case is when the user is challenged to provide his password, while in other cases certain mathematical operations may be performed, such as hash functions, which utilize the shared secret, e.g. the message authentication code. In most case, though, when one refers to authentication based on something the user knows, we usually refer to passwords or PINs are implied. The inherent weakness of the aforementioned schemes is rather obvious: the human limitation to remember long random strings of characters that would formulate a strong password.

Intense research work has taken place in order to evaluate and improve password schemes. Nevertheless, weak passwords are very often chosen by users, which are subsequently taken into advantage by malicious attackers to gain unauthorized access to information systems. Graphical passwords provide a promising alternative to the traditional text-based password authentication. The key concept is the human ability to recognize or recall images more easily than textual information [1]. Text-based passwords remain the dominant authentication scheme due to their ease of use, inexpensive deployment, user familiarity and acceptance contrary to the more demanding requirements of other authentication schemes, like the costs involved in token-based authentication, privacy and user acceptance issues in biometrics, etc.

A successful security mechanism has to be both usable and secure. A security mechanism that is not user-friendly is inherently insecure, since users will systematically try to misuse it or bypass it altogether. Graphical passwords have to address a critical challenge: the balance between the security of the produced graphical password and the user friendliness and acceptance. In this paper, we present a novel graphical password scheme, namely Novel Authentication with Visual Information (NAVI), which is meant to be a graphical password scheme that is easy to implement and user-friendly while providing a high level of assurance in terms of password strength. Our goal was to develop a scheme that produces passwords from a adequately large keyspace, where the password keyspace is defined as the set of different possible values a password can take, and that relies on existing widely-used services that are easy to use and familiar to end users.

The remainder of this paper is organized as follows: In section 2 we provide an overview of some related work, while in section 3 the proposed scheme is described. A security discussion follows in section 4, where issues regarding the strength of both text-based and graphical passwords are briefly analyzed and a comparison of the aforementioned schemes is provided. Finally, in section 6 a proof of concept implementation is presented and section 7 provides conclusions and directions for future work.

II. RELATED WORK

Graphical passwords were first introduced by G. Blonder in 1996 [2]. Since then, a variety of graphical authentication schemes have been proposed and implemented and some commercial products have been launched. Nevertheless, none of them has been widely adopted or can be considered practical to use and secure at the same time.

Graphical passwords are usually formulated by a user by selecting a set of images in a specific order or drawing. In the context of this paper the term graphical passwords will be used in the broadest possible sense to include every password scheme that authenticates users based on visual information, including the selections of images, the clicking or dragging on images, drawing etc.

One way to categorize graphical password systems is to separate them in two broad categories:

- recognition-based , and
- recall-based.

In a recognition-based authentication scheme, the user is presented with a set of images and he is then requested to recognize the set of images he originally had selected as his password at the registration stage.

Passfaces [3] is a commercial product where the user authenticates himself by selecting from a grid of nine faces a previously seen human face. The process is repeated until all previously (typically three to seven) seen faces have been identified. In Awase [4] a user uploads a picture of his choice, and, then, he is requested to recognize it among a number of decoy images. "Use your Illusion" [5] has a similar approach but also takes into account the difficulties that may be encountered when using a low resolution device and pictureshave been blurred in a controlled manner.

In "Déjà vu" [6] users are requested to select 5 pictures from a set of 25 images in order to create their graphical password. At the same time the users are requested to create a traditional password.

Sobrado and Birget [7] suggested a number of graphical passwords schemes that are resistant to shoulder surfing attacks. The user selects 3 images from a large set. In order to be authenticated the user needs to select a point in the triangle the 3 images formulate. Asghar et al. [8] presented a cryptanalysis attack that greatly undermines the overall security of this scheme.

In recall-based authentication schemes, a single picture is displayed to the user, who in turn selects / clicks / drags / draws one or more locations on the picture to initialize his password at the registration stage. At the log in phase, the user will be requested to reproduce these operations. Since it is expected that the users will be unable to reproduce their drawings with complete accuracy, an error threshold must be set.

A commercial product designed specifically for mobile devices is SFR Password, where the user enters the previously defined spots in the correct order to unlock the mobile device [9]. Input precision is a problem with this technique, which limits the possible password keyspace if the tolerance is high and may be impractical if the tolerance is too low.

In Passpoints [10] the user credentials are a sequence of clicks in positions in given image positions. Image processing techniques, a biased selection from users and other attacks have been demonstrated against such a system that undermine its security and render it effectively broken [11], [12].

In Draw A Secret (DAS), introduced by Jermyn et al. [13], the user has to draw something in a two dimensional grid. The coordinates of the grids are used to formulate the password. DAS offers an adequate keyspace.

Three dimensional objects and the environment can also be utilized to formulate a password scheme [14], [15]. The sequence of actions and the interactions with objects in this environment formulate the 3-D password. 3-D passwords are considered a multifactor authentication scheme that encompasses graphical passwords.

For a more comprehensive overview of existing literature and a further analysis of the usability and security issues of graphical passwords, surveys are available in Dunphy et al. [16], Biddle et al. [17], Renaud [18], Hagiz et al. [19], Monrose et al. [20] and Suo et al. [21].

III. Novel Graphical Password Scheme

A. Overview of NAVI Authentication Scheme

In this paper, we introduce a novel knowledge-based authentication scheme that belongs to the recall-based graphical passwords family. The credentials of the user are a route (or multiple routes) of his choice. For simplicity, we will only discuss the case where a single route is required for authentication.

The route is designed in a predetermined map. The user selects the starting and the ending point and the route is calculated by the provider of the route planning service. Further details with regards to the specifics of the route selection are discussed in the section that follows. Furthermore tin order to highlight the feasibility of the authentication mechanisms proposed, a case study with a prototype implementation of the proposed scheme based on the Google mapsTM mapping service will be presented (Figure 1).

Note that in principle our scheme can be implemented in any other platform that supports maps and allows users to plan routes. Being based on Google maps provides flexibility ease of use and user acceptance, since Google maps is a widely-used service and there is a large pool of users already accustomed to it. Assuming a Google map approach. the routes will be two dimensional (Google maps utilize a variation of the Mercator projection, a widely-used method for maps projection that takes its name from its inventor Gerardus Mercator, a Flemish mapmaker who lived in the 15th century).

The user is challenged to choose a username and a route of his choice the first time he logs into the system (the equivalent of setting the password). This pair will be called "route password" from now on. Whenever the user wishes to log in, he is challenged to recreate the route he has chosen as the route password. If the route matches the route password the user is successfully authenticated and he is allowed to log in.

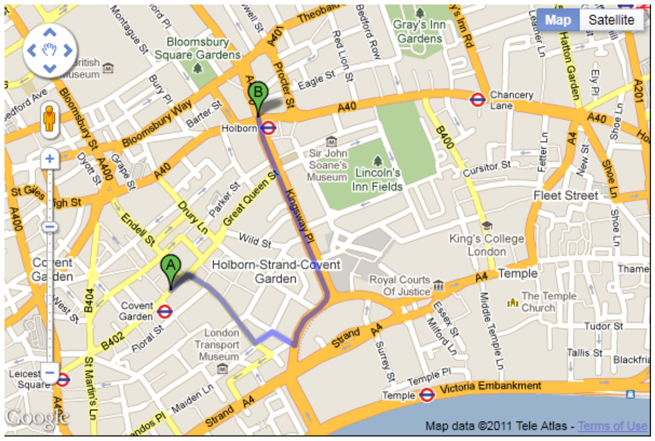

Figure 1. Route in Google maps.

B. Requirements for a Secure Route

Since a route is something the user knows, it may be subject to attacks similar to those in other recall-based schemes. A set of criteria are proposed in order to strengthen the route password, the equivalent of setting up a strong password. As such, the route must be unpredictable, so that the starting and ending points as well as the intermediate route cannot be guessed. Furthermore, the route should be long enough in order to include many turns in order to provide adequate complexity. It is also necessary that the user does not use the predefined route suggested but he modifies it by introducing at least one deviation from the route created by Google maps. To be more specific the following user guidelines can be provided:

- The starting and ending points must be unpredictable, e.g. they should not include home or work addresses, and locations like a user's gym and the like should be avoided.
- The route must include a minimum of 4 turns (in other words the route must consist of at least 6 points).
- At least one deviation from the automatically calculated route must be introduced.
- At least one route should be located in a country different than the home country of the user.

It is important to take into account that the first requirement cannot be strictly enforced, since it relies upon the user, and there is no generic way to confirm that the user has not used, for example, his home location as the end point of his route, just like no system can ensure that the user has not used his license plate as part of his password. On the contrary, the remaining requirements can be strictly enforced by the authentication system and they can be parameterized, if needed, by the system administrator and the security policy applicable to the particular system / application.

We shall shortly illustrate that the strength of the produced secret mainly relies on the number of "turns" and their unpredictability. The other parameters are useful in order to counter attacks that are based on the knowledge of information regarding the user.

IV. Encoding the Route Password

It not straightforward what will be the exact string of characters that the system will require for authentication. We will discuss two approaches with some variations and their corresponding features. Namely, the following solutions will be examined:

- using the coordinates with a minimum of processing, and
- approximate the route with a polynomial.

A. Minimum Processing

In Google maps every turn, which is equivalent to a point, is represented with a pair of numbers e.g. (51.50364050493053, -0.135955810546875). From the 16 digits of each coordinate we will take into account only the decimal numbers, and exclude the integer values. The reason for that is that the integer values can be subject to guessing attacks. The range of the decimal values are limited by the fact that a significant set of them corresponds to uninhabited areas where it would be impossible for a user to plan a route, e.g. in oceans, mountains etc. We also exclude the first two decimals as they provide poor entropy, since unless the user chooses a very long route these digits will remain constant and even if they increase or decrease, this value it will be one. Furthermore, since the first two digits define a large geographical area it is expected that they will be susceptible to guessing attacks. The remaining 12 digits can be used to formulate the secret.

For every turn included in the route, 24 digits of "secret" can be produced. In order to deal with guessing attacks it would be preferable to perform a one way "function", i.e. perform an XOR (logical AND) in order to increase the resistance to attacks but this will result in limiting the size of the secret to 12 decimal digits per turn. Assuming that the route includes 4 turns, our secret can be composed of 4*12=48 digits. The password keyspace of the produced secret will be $10^{48} \approx 2^{160}$.

If no processing takes place, the first and the last points of the route cannot be utilized since they are subject to an accuracy error, as they are manually selected the user. Alternatively, a threshold can be set for every point to be acceptable and not to be considered a user error. If that is the case, the logical AND operation cannot be performed and the coordinates must be used as is. Furthermore, in the case of Google maps (and in principle in any third party service that is not under a user's control) changes may occur to the coordinates of a point, e.g. they can be altered in order to provide better accuracy.

Encoding the secret with minimum processing allows for a large keyspace, easy implementation and it also introduces minimum overhead to the client. However, the aforementioned solutions have a very strict requirement in terms of the precision of the produced secret. A single deviation in one digit will result in a change of the route password. A threshold for accuracy can also be set in the case where no processing takes place.

B. Polynomial Fitting

The route presented in Fig. 1 can be also represented as a set of points in a 2-dimensional projection as depicted in Fig. 2, where each turn is presented as a point, independent of t the sequence.

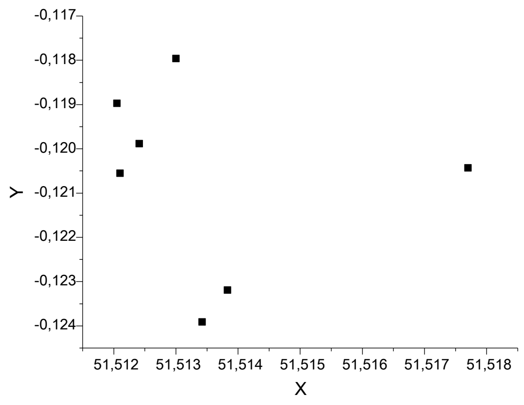

Figure 2. Route represented as points.

This set of points can be fitted by a polynomial. The polynomial can be defined by utilizing the least squares approximation and/or variations of it, like the constrained or weighted least squares approximation.

1) Least Squares Approximation

Given a set of data (x_1, y_1), (x_2, y_2), … (x_n, y_n), using the least-squares method we can produce an mth degree polynomial y(x) (or a line or a parabola, but we will examine the most generic solution) where n>m:

$$y = a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m \quad (1)$$

The best fitting curve y has the least square error, or equivalently one wants to minimize the error $\prod$ ::

$$\prod = \sum_{i=1}^{n} [y_i - f(x_i)]^2 = \sum_{i=1}^{n} [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m)] \quad (2)$$

where y is the fitting curve and $f(x_i)$ are the actual values provided.

The unknown coefficients a_0, a_1,... , a_m can be obtained by solving the following linear equations:

$$\sum_{i=1}^{n} y_i = \alpha_0 \sum_{i=1}^{n} 1 + \alpha_1 \sum_{i=1}^{n} x_i + \alpha_2 \sum_{i=1}^{n} x_i^2 + ... + a_m \sum_{i=1}^{n} x_i^m$$

$$\sum_{i=1}^{n} x_i y_i = \alpha_0 \sum_{i=1}^{n} x_i + \alpha_1 \sum_{i=1}^{n} x_i^2 + \alpha_2 \sum_{i=1}^{n} x_i^3 + ... + a_m \sum_{i=1}^{n} x_i^{m+1}$$

$$\sum_{i=1}^{n} x_i^2 y_i = \alpha_0 \sum_{i=1}^{n} x_i^2 + \alpha_1 \sum_{i=1}^{n} x_i^3 + \alpha_2 \sum_{i=1}^{n} x_i^4 + ... + a_m \sum_{i=1}^{n} x_i^{m+1}$$

...

$$\sum_{i=1}^{n} x_i^m y_i = \alpha_0 \sum_{i=1}^{n} x_i^m + \alpha_1 \sum_{i=1}^{n} x_i^{m+1} + \alpha_2 \sum_{i=1}^{n} x_i^{m+2} + ... + a_m \sum_{i=1}^{n} x_i^{2m} \quad (3)$$

The route from Covent Garden to Holborn that serves as our example (see Fig. 1) consists of the starting point, the ending point and 5 turns. This specific route can be fully represented by the information in table I, where each point is represented by a set of coordinates (Latitude, Longitude) as utilized by Google maps:

In order to address the already mentioned security concerns with regards to the entropy of the integer values provided, we will discard them. Now the starting point, the ending point and the turns of the route have been transformed as depicted in Table I

TABLE I. TURNS WITHOUT INTEGER VALUES

Point	X-axis	Y-axis
Starting Point	0,51342	-0,123910000000023
Turn 1	0,51383	-0,123190000000022
Turn 2	0,5121	-0,120549999999980
Turn 3	0,51241	-0,119879999999966
Turn 4	0,51205	-0,118969999999990
Turn 5	0,5130	-0,117960000000039
Ending Point	0,5177	-0,120429999999942

We will perform the polynomial fitting using the least squares approximation to find a polynomial of 4th degree that fits the points presented in Table I. Fig. 3 presents the polynomial graphically. The coefficients of the polynomial are calculated by solving the set of equations presented above in (3):

TABLE II. COEFFICIENTS OF THE 4TH DEGREE POLYNOMIAL WITH SEVEN POINTS

Coefficient	Value
α_0	$-1,98164*10^7$
α_1	$1,54002*10^8$
α_2	$-4,48805*10^8$
α_3	$5,81302*10^8$
α_4	$-2,82342*10^8$

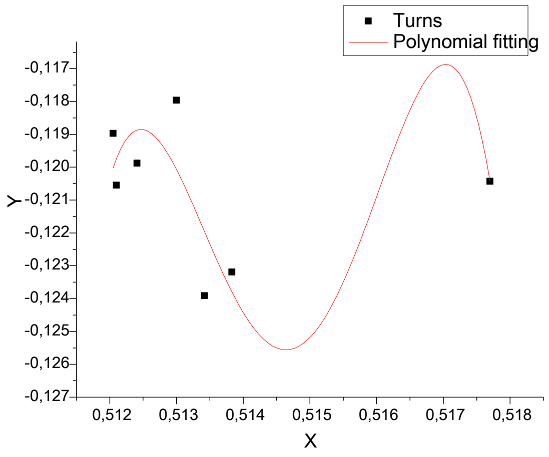

Figure 3. Polynomial fitting for all points.

As mentioned before, the starting and ending points are subject to accuracy errors, so if we wish to avoid introducing an error acceptance threshold we can instead ignore these points and utilize the remaining turning points. We will use polynomial fitting to find a polynomial of 4th degree that fits the points presented in Table I, excluding the starting and ending points. Again, by solving (3) we calulate the coefficients of the polynomial (see Table III). The polynomial is graphicaly presented in Fig. 4.

TABLE III. COEFFICIENTS OF THE 4TH DEGREE POLYNOMIAL WITH FIVE POINTS

Coefficient	Value
α'_0	3,61167*109
α'_1	-2,81691*1010
α'_2	8,238881010
α'_3	-1,070981011
α'_4	5,220651010

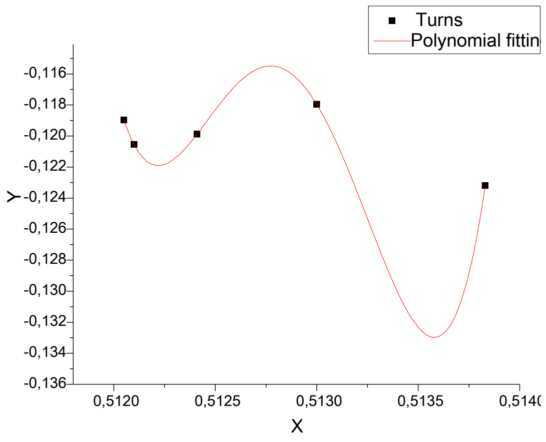

Figure 4. Polynomial fitting for the turns.

In any case, the generated secret is the function:

$$f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 \qquad (4)$$

For simplicity's sake, the graphical password could be the values of the coefficients themselves. The key space in this case would not produced in a straightforward manner, since intense mathematical operations would be necessary. The produced keyspace will be subject to limitations due to the fact that floating point numbers, in a programming language, have limited precision We assume that the calculations will be performed in a web-based environment using JavaScript where numbers are represented using 64 bits. In that case we would have a keyspace of $5*2^{64} \approx 10^{20}$.

In the case of Google maps the use of polynomials can offer a very significant advantage: in case minor modifications in the precision of certain points in the map appear, we can set a threshold in the precision of the resulting polynomial and, thus, any minor modification in the precision will be absorbed. Updating the password file with every possible modification besides being a cumbersome task may result in introducing vulnerabilities that can be exploited by attackers. If T is the threshold value, then a user that is challenged with x_1 in order to authenticate himself should return a value in the range $A' = f(x_1) \pm T$, resulting ina reduced keyspace of $2^{64}/(2*T)$.

2) Least Square Approximation with Weights and Constraints

The starting and ending points of the route are not only subject to accuracy errors when the user is selecting them, but changes may also occur to the rest of the turns due to changes introduced by the provider of the map service. So far, in order to cope with this issue we ignored these facts to retain the generated password accuracy. In the polynomial encoding approach, we can utilize the Least Square Approximation with weights. Then, Equation (1) becomes:

$$\prod = \sum_{i=1}^{n} w_i [y_i - f(x_i)]^2 = \sum_{i=1}^{n} w_i [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \ldots + a_m x_i^m)] \qquad (5)$$

where $w_i>0$ are positive weights that allow placing more emphasis on reliable data points and less on the others by choosing for them larger and smaller values respectively. In the least square approximation with constraints, constraints can be imposed, for example, to require that at certain points the error will be zero or an acceptable threshold of deviation can be set.

V. SECURITY PERFORMANCE

A. Brute force and Dictionary Attacks

The keyspace is defined as the set of different possible values of a key. A password with n characters, where each of those characters can have c different values will have a keyspace size $k = c^n$. A large keyspace provides assurance against brute force attacks, where the attacker is exhaustively trying all possible values of the password to be cracked. However, a large keyspace does not guarantee the security of an authentication scheme. It is often the case where an attacker instead of trying to check by brute force the entire keyspace will select sub-keyspaces that fulfill certain criteria that render them as a more likely selection by users. This is often called a dictionary attack.

For example, if we take a five digit password consisting of upper and/or lower case letters from the English alphabet the keyspace size S is $52*52*52*52*52 \approx 4*10^8$. If the password is generated truly randomly and an attacker has launched a brute force attack he would have $1/4*10^8$ chances that any single guess would match a given password.

However, if a user chooses his own password he would choose an easy-to-remember password rather than a random one. Let us assume that the user chooses a word from the English language. The exact number of the English words is nearly impossible to define accurately, and it is equally hard to estimate the words a human individual can remember. However, it is hard to argue that a human will be capable of memorizing and using more than 200.000 words, even in an over-optimistic scenario where the end user is a genius. In this case the attackers will have a 1/200.000 chance that any single guess would match the given password. An attacker that has some additional information of his target can launch a more targeted guessing attack that will consist of words like the target's name, maiden name, city of birth, country, city and address of residence etc.

The subset of English words is a subset of the entire password keyspace. Any subset of the password keyspace S that allows an attacker to guess passwords with less effort is called a Weak Key Sub-space, WS, where $WS \subseteq S$. Dictionary attacks are based upon weak password subspaces. Essentially, a dictionary is a Weak Key subspace, like the set of English words mentioned before. In another real-life example, for a network router a weak subspace consists of the default passwords as initialized by the related vendor.

Figure 5. Password keyspace and weak key sub-spaces.

The most important feature of a dictionary is its size. We must also take into account the effort required to create a dictionary. It is often the case where the cost of generating a dictionary is ignored altogether since in most cases one precomputation is adequate to create a dictionary and that cost is considered negligible. However, in some cases the precomputation of dictionaries is not feasible, e.g. if the pictures to appear are not anticipated, then a dictionary has to be created on the fly. In such cases, the cost of generating a dictionary may be as important as the size of the dictionary [22].

B. Text Passwords vs. Graphical Passwords

Text passwords have been exhaustively analyzed and attacked as they are the dominant user authentication method used. On the contrary, limited research has taken place towards attacking and cracking graphical passwords since they have not been used for such a long period and they have not yet been widely adopted. Nevertheless, graphical password schemes appear to suffer from problems similar to text passwords, like dictionary attacks.

The following criteria will be used in order to examine and evaluate the proposed solution against text-based passwords and other graphical password schemes.

- Security
  - Brute force attacks
  - Dictionary attacks (guessing attacks and statistical attacks)
  - Spyware and keyloggers
- Non technical hacking
  - Social engineering
  - Shoulder surfing

In order to effectively counter brute force attacks a large password keyspace is required. As we have shown, NAVI enables the selection of passwords with an efficiently large keyspace. Recognition-based schemes offer a limited key space, rendering them vulnerable to brute force attacks. In order to increase the key space the number of pictures can be increased and/or the user be challenged several times. Apparently, such an approach will be time-consuming, tiresome and also hard to remember.

Countering dictionary attacks is neither easy nor straightforward, as it depends to a large extent on the end user. Dictionary attacks have proven to be very effective against text-based passwords [1], and a series of automated tools are available for that purpose, e.g. brutus, and John the Ripper. Graphical passwords have the advantage that they are harder, although by no means impossible to launch automated attacks against them. Although graphical passwords have not been exhaustively attacked and investigated, significant work towards realizing effective attacks has been illustrated. Dictionary attacks can also be launched against graphical passwords, since the idea that some sets of graphical passwords are more likely to be chosen than others is applicable in essentially any knowledge-based scheme. These sets naturally define a weak key subspace and the related graphical dictionary. Such automated attacks have been demonstrated to be effective against PassPoints in Dirik et al. [11], Thorpe et al. [12] and Oorschot et al. [22].

We argue that NAVI has an increased resistance to dictionary attacks due to the unpredictability of the "picture" the user selects. Launching a dictionary attack on NAVI will require significant effort, since the picture/map to be attacked is unknown to the attacker. If the location of the map was known, then places of interest, crossroads etc, would be identified and used to formulate a dictionary. However, this assumes that the attacker has knowledge of the area the user has selected, which must be acquired through other means, such as social engineering. The attacker cannot be aware of which part of the map a user will use but he can only make an educated guess of a larger area in case he has some additional information regarding the user. Nevertheless, even in the case we assume that dictionary attacks are feasible against NAVI, the attacker will also have to be able to handle the overhead of creating the dictionary in real time, prior to launching his attack. Even in that case, the attacker should not only guess the starting and ending points but also the deviation from the default route. This renders the creation of dictionaries a cumbersome task, since they must be created on the fly imposing a significant overhead on the attacker.

Guessing attacks can be seen as a more targeted sub-category of the dictionary attacks. An effective and provable defense against such attacks is nearly impossible, since the selection of passwords ultimately relies on the end users, and the attempts to "force" users into choosing secure passwords produce questionable results. Recall-based schemes have the advantage of rendering such attacks more cumbersome for the malicious user since the attacker will have to create his dictionary on demand.

Spyware/key loggers have the ability to log the keystrokes and hence the input of a text-based password. More advanced spyware has also the ability to take screenshots that can compromise graphical passwords too. In case of text-based password end-point security, e.g. antivirus, spyware detection software, appears as the only solution. Graphical passwords even though not immune to such attacks are more resistant due to their very nature.

When it comes to shoulder surfing, the nature of graphical passwords is a handicap, and apparently they are more vulnerable to such attacks. In the following section we suggest a solution that mitigates the risk of shoulder surfing for NAVI by utilizing opacity and, thus, limiting the range at which a shoulder surfing attack can be successful.

Social engineering requires little or no technical skills and similarly to guessing attacks, a defense against such attacks mainly relies on the user, making it hard, if not impossible, to defend by implementing solely technical countermeasures.

In Table IV, we summarize the conclusions of the above discussion with our estimation of the NAVI performance against the attacks aforementioned.

TABLE IV. SUMMARY OF NAVI SECURITY FEATURES

Attack method	Security element	NAVI performance
Brute force	Password keyspace	■■■■■ (Very High)
Dictionary	Weak Password keyspaces	■■■■■ (Very High)
Guessing	Recall or recognition capabilities of end users	■■■□□ (Medium)
Spyware	End point security / Data input method	■■■■□ (High)
Shoulder surfing	Data input method	■□□□□ (Very Low)
Social engineering	Credentials type	■■■□□ (Medium)

Furthermore, following the security discussion, in Table V we summarize the comparison of NAVI with graphical-based password schemes (recognition and recall-based) and text-based passwords.

TABLE V. PASSWORDS SCHEMES COMPARISON

Attack method	NAVI	Recognition based	Recall based	Text based
Brute force	■■■■■	■□□□□	■■■■□	■■■□□
Dictionary	■■■■■	■□□□□	■■■■□	■■□□□
Guessing	■■■□□	■■□□□	■■■□□	■■□□□
Spyware	■■■■□	■■■■□	■■■■□	■■□□□
Shoulder surfing	■■□□□	■■□□□	■■□□□	■■■■□
Social engineering	■■■□□	■■■□□	■■■□□	■■■□□

C. Countering Shoulder Surfing

The security analysis illustrated that the proposed scheme has the same or better performance compared to text-based passwords and graphical password schemes, with the exception of shoulder surfing attacks. In the following section we discuss techniques to enhance the resistance of NAVI to such attacks. We have identified the following possible solutions:

- **Opacity** is the extent to which something blocks light. By covering the map with the route, for example, with a blue picture with opacity of 50%, the route will be behind a "blue certain" thus visible to the user that is close to the screen but not visible to an interceptor a few meters away. For more details see the next section.
- **Multiple Routes**. Have a pool of routes created by users; each time the user will be challenged with a different one chosen randomly.
- **Partial use of route** Instead of creating the full route the user selects, for example,4 points, which are verified to belong to the route to formulate the password. The shoulder surfing attack still provides information to the attacker since he limits his attack paths in the context of a specific area that he has observed. When the user selects/clicks on a point on the route, this point may not appear on the screen. In the latter case there is an obvious reduction in the password keyspace.

VI. PROOF OF CONCEPT

In order to illustrate the feasibility of the proposed scheme a prototype has been implemented. JavaScript and the Google maps API have been used for the implementation. In addition to the elements already discussed the implementation offers the following features.

In order to add further complexity the end user can select whether his route will be calculated for driving or walking, hence although the starting and ending points maybe the same, the route will be altered. Similarly, the user is offered the choice of a calculated route that avoids highways and/or tolls. Finally, the user is forced to define one more point that the route has to pass through (see Fig. 6). Furthermore, in Fig. 8, the use of opacity in order to counter shoulder surfing attacks is presented.

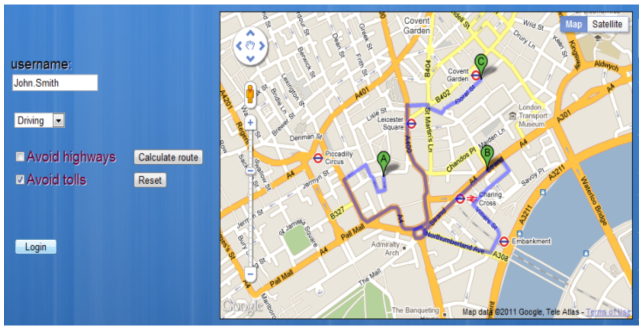

Figure 6. Planning the route password.

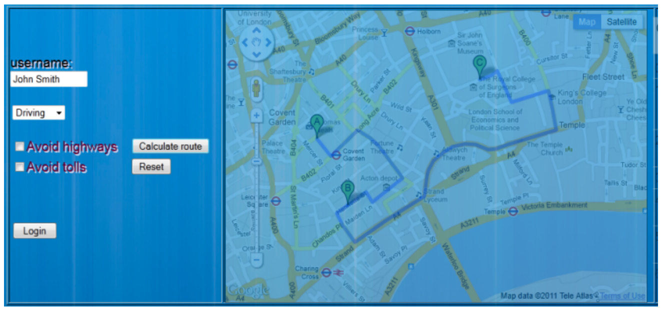

Figure 7. Google map with opacity.

VII. CONCLUSIONS

This paper presented NAVI, a novel graphical password scheme where the credentials of the user are a route designed on a predetermined map. We believe that NAVI is the first model to combine strong graphical passwords along with ease-of-use. However, the strong security features remain to be more strictly proven. Future work involves examining the usability of the scheme through the deployment of the system and its usage by a number of users. The proof of concept presented illustrated the feasibility of implementation and the possibility of deployment of the proposed scheme in web applications without the need to significantly modify the underlying applications. The user testing will provide valuable feedback with regards to the usability and viability of the authentication system and will try to answer the following key questions: Is it easy to use (Login time)? Is it easy to memorize the routes? Is it easy to learn and accept the use of the system?

REFERENCES

[1] C. Kuo, S. Romanosky, and L. Cranor 2006. Human selection of mnemonic phrase-based passwords. In Proceedings of the second symposium on Usable privacy and security (SOUPS '06). ACM, New York, NY, USA, 67-78. doi:10.1145/1143120.1143129

[2] G. Blonder, Graphical Password. In Lucent Technologies, Inc., Murray Hill, NJ, United States Patent 5559961, 1996.

[3] Real User Corporation, PassfacesTM, http//:www.realuser.com.

[4] T. Takada, T. Onuki, and H. Koike 2006. Awase-e: Recognition-based image authentication scheme using users' personal photographs. In Innovations in Information Technology. 2006 , vol., no., pp.1-5, Nov. 2006 doi:10.1109/INNOVATIONS.2006.301970

[5] E. Hayashi, R. Dhamija, N. Christin, and A. Perrig 2008. Use your illusion: secure authentication usable anywhere. In Proceedings of the 4th symposium on Usable privacy and security (SOUPS '08). ACM, New York, NY, USA, 35-45. doi:10.1145/1408664.1408670

[6] R. Dhamija, A. Perrig, 2000. Déjà Vu: a user study using images for authentication, In Proceedings of the 9th Conference on USENIX Security Symposium - Volume 9 (SSYM'00), Vol. 9. USENIX Association, Berkeley, CA, USA, 4-4.

[7] L. Sobrado and J. Birget, 2002. Graphical Passwords. The Rutgers Scholar , An Electronic Bulletin of Undergraduate Research, Rutgers University, New Jersey, Vol. 4 (2002)

[8] H. J. Asghar, S. Li, J. Pieprzyk, and H. Wang, 2010. Cryptanalysis of the convex hull click human identification protocol. In Proceedings of the 13th International Conference on Information security (ISC'10),. Springer-Verlag, Berlin, Heidelberg, 24-30. doi: 10.1007/978-3-642-18178-8_3

[9] SFR IT - Engineering,
http://www.sfr-software.de/cms/EN/pocketpc/sfr-password/#.

[10] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, 2005. PassPoints: design and longitudinal evaluation of a graphical password system. Int. J. Hum.-Comput. Stud. 63, 1-2 (July 2005), 102-127. DOI=http://dx.doi.org/10.1016/j.ijhcs.2005.04.010.

[11] A. E. Dirik, N. Memon, and J.-C.Birget, 2007. Modeling user choice in the PassPoints graphical password scheme. In Proceedings of the 3rd Symposium on Usable privacy and security (SOUPS '07). ACM, New York, NY, USA, 20-28. DOI=http://doi.acm.org/10.1145/1280680.1280684

[12] J. Thorpeand and P. V. Oorschott , 2007. Human-seeded attacks and exploiting hot-spots in graphical passwords. In Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium (SS'07), Niels Provos (Ed.). USENIX Association, Berkeley, CA, USA, , Article 8, 16 pages.

[13] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter and A. D. Rubin, 1999. The design and analysis of graphical passwords. In Proceedings of the 8th Conference on USENIX Security Symposium - Volume 8 (SSYM'99), Vol. 8. USENIX Association, Berkeley, CA, USA, 1-1.

[14] F. A. Alsulaiman and A. El Saddik, 2008. Three-Dimensional Password for More Secure Authentication. In IEEE Transactions on Instrumentation and Measurement, , vol.57, no.9, pp.1929-1938, Sept. 2008. doi:10.1109/TIM.2008.919905

[15] V. Mhaske-Dhamdhere, G. A. Patil, 2010. Three Diamentional Object Used for Data Security. In International Conference on Computational Intelligence and Communication Networks (CICN), 2010, pp.403-408, 26-28 Nov. 2010. doi: 10.1109/CICN.2010.83

[16] P. Dunphy, A. P. Heiner, and N. Asokan 2010. A closer look at recognition-based graphical passwords on mobile devices. In Proceedings of the Sixth Symposium on Usable Privacy and Security (SOUPS '10). ACM, New York, NY, USA, , Article 3 , 12 pages. doi:10.1145/1837110.1837114

[17] R. Biddle, S. Chiasson and P.C. Oorschot Version: October 2, 2009. Technical Report TR-09-09, School of Computer Science, Carleton University, Ottawa, Canada.

[18] K. Renaud, 2009. Guidelines for designing graphical authentication mechanism interfaces. In Int. J. Inf. Comput. Secur. 3, 1 (June 2009), pp. 60-85. doi:10.1504/IJICS.2009.026621

[19] M. D. Hafiz, A. H. Abdullah, N. Ithnin, and H. K. Mammi, 2008. Towards identifying usability and security features of graphical password in knowledge based authentication technique. In Modeling

[19] & Simulation, 2008. AICMS 08. Second Asia International Conference on , vol., no., pp.396-403, 13-15 May 2008. doi:10.1109/AMS.2008.136

[20] F. Monrose and M. Reiter, 2005. Graphical passwords. In Security and Usability: Designing Secure Systems That People Can Use, L. Cranor and S. Garfinkel, Eds. O'Reilly Media, 2005, ch. 9, pp. 157–174.

[21] X. Suo, Y. Zhu, and G. Owen, 2005. Graphical passwords: A survey. In Annual Computer Security Applications Conference (ACSAC'05), Dec. 2005. doi:10.1109/CSAC.2005.27

[22] P.C. Oorschot, A. Salehi-Abari and J. Thorpe, 2010. Purely Automated Attacks on PassPoints-Style Graphical Passwords. In IEEE Transactions on Information Forensics and Security, vol.5, no.3, pp.393-405, Sept. 2010, pp. 393 – 405, 2010. doi: 10.1109/TIFS.2010.2053706