Instituto Superior Técnico
Cryptography and Network Security
Author: Miguel Alves Ferreira, 75471
Email: miguel.alves.ferreira@ist.utl.pt,

Semester Project: Graphical Passwords for User Authentication

1 Introduction

The authentication process from a partie A to a partie B consists of an interaction such that party A provides an attribute of his to party B which comproved the identity of A to B. In computer based systems, the authentication process takes the form either a machine-machine interaction and human-machine interaction. The first, The second, also named as **User authentication**, in particular, consists of an interaction where a user given input is provided to the machine so to comproved that user U is allowed access in the given system.

In the context of User Authentication, PIN and password based Authentication are widely regarded as the most commom methods to accomplish user authentication to a computer based system. PIN and password based Authentication consist of a scheme where a user proviedes an input consisting of a user identifier (also known as username) and respective textual password; such inputs (or a function of it) is passed to the machine which stores the set of all user identifies, confirming or not its pressence. Examples range from web applications, to ATM authentication. The security of password based Authentication relies on the following permisses:

1. The password strength is a function of its length, its complexity (measured by the entropy) and its unpredictability (measured by the).
2. The host storing the password does not store in plain text but encrypted using a cryptographafic hash function.
3. Password authentication does not normally require complicated or robust hardware since authentication of this type is in general simple and does not require much processing power.

A secure password based Authentication is naturally yield by a password which complies with the notions pointed out above. However, User Authentification is a human. Since a password scheme implies a precise recall of textual input, users tend to select either short passwords or passwords that are easy to remember [] []. Hence, the human factors of the authetification scheme are open fundemental , which is explored by a wide range ofof attacks.

1. Brute Force Attacks -
2. Dictionary Attacks - (Online and Offline)
3. Social Engineering Attacks -
4. Shoulder Surffing -

Cheswick and Bellovin have pointed out that weak passwords are the most common cause for system break-ins []

Alternatives consist of

Graphical Password come up has an alternative to.

2 Graphical Passwords

Several examples of graphical passwords can be found in the literature.

2.1 Sobrado and Birget in [?]

2.2 Passface, by in [?]

2.3 Passpoint, by in [?]

2.4 DAS - Draw a Secret, by Jermyn and all. in [?]

DAS — Draw a Secret — consists of a graphical password scheme where a user's drawing on a screen is associated with the user's password. The screen is seen as a two dimensional grid such that each cell is mapped to a binary string. As a matter of example, consider the mapping and drawings exibited in Figure **??**. It is argued that splitting the display screen in something as small as a 5×5 grid yields a passwords space which as larger than the most commom textual password spaces.

Such technique, however, has some backdrops. Firstly, there is a clear design problem regarding the cell boarder and its interaction with the users drawings, *i.e.*, drawings whose lines slightly cross cell boarder when they are not supposed to. Such problem, however, can be surpassed by considering an approach similar to the one proposed for *Passpoint*, where the password inputed upon registration is weigthed against the user's original password. Secondly, there is an inherant need for an user interface which allows the user to type its password. Besides, Thorpe and van Oorschot in [**?**] have anaylised the passwords space yield by the DAS technique and concluded that while this scheme is less susceptible to dictionary attacks, when compared to textual passwords, the graphical password is potentically much smaller, under the assumption that user tend to choose mirror symmetric passwords.

3 GPAPI

According to the discussion conducted in the previous section, we conclude that a secure but usable (from a user prespective) Graphical Password system for User Authentication must respect the following permisses.

1. The system should be based on recognition and not on recall, *i.e.*, during the authentication process, the user should not provide a graphical password but instead identify its graphical password (or a subset of it) from a set provided by the system.

2. The system must guarrantee that the user chosen graphical passwords are strong, *i.e.*, (a) The system must guarrantee that the passwords length is large enough; (b) The system must guarrantee that images making up the graphical passwords are select independently (c) The system must guarrantee that the graphical passwords are uniformly distributted over the graphical passwords space.

3. The system must provide a graphical password alphabet composed of images which are not only perceptually distinguishable and user identifiable [1], by also difficult to recompute for a attacker which does not have the knownledge of system inner state.

4. The system must be resistant to Intersection, Shoulder Surfing and Social Engineering Attacks.

[1]

In order to achieve the requirement mentioned above, and in line with the work of Dhamija and Perrig in [**?**], we propose a Graphical Password system named GPAPI. It outline as follows. The system stores in a memory a set of M images generated according to the Random Art Algorithm [2], corresponding to the Graphical Password alphabet. A user registrates in the system by creating a portflio of N ordered images selected from a set of M possible images provided by the system, leading to a password of length N. Upon registration, a user must complete a challenge proposed by the system to achieve authentication. The challenge envolves a set of m images, composed of n images belonging to the user's portfolio and $n - m$ images not belonging to the user's portfolio. The user solves the challenge by correctly identifying all n images present in its portfolio and post them in their relative order.

3.1 Graphical Password Alphabet Generation

The requirements imposed on the generation of the graphical password alphabets ressemble the requirements for hash visualization algorithms, as defined by Perrig and Song in [**?**]. A hash visualization algorithm, building on top of the definition of hash functions, is defined as follows.

Definition 1. *(Hash Visualization Algorithm) A Hash Visualization Algorithm describes a function $h : x \rightarrow I$ mapping an arbitrary string x to an image I which satisfies the following properties.*

1. ***Ease of Computation:*** *Given h and an arbitrary string x, $I = h(x)$ must be easy to compute.*
2. ***Near-one-way Property:*** *Let the relationship $I_1 \simeq I_2$ denote the fact that two images I_1 and I_2 are perceptually indistinguishable. (a) For any image I, it is computationally infeasable to find an arbitraty input string x such that $I \simeq h(x)$. (b) For any input string x, it is computationally infeasable to find an arbitraty input string x' such that $h(x) \simeq h(x')$.*
3. ***Regularity Property:*** *Let I be any image and let $\mathcal{F}\{I\}$ be its Fourier Transform. We define an image to be regular if, for some fixed $\delta$and ϵ, $\sum_{f>\delta} |\mathcal{F}\{I\}(f)| < \epsilon$*

As mentioned, notice the existing matching between 1. The perceptually distinguishability between images and the *Near-one-way Property*; 2. The user identifiability of images and the *Regularity Property* (since humans are particularly sensible to images with shapes when compared to images with a great deal of white noise) 3. The possibility for uniformly distributed and independently select images associated with the fact the TOOODOOOO.

Due to the consideration above, we let our Graphical Password alphabet consist of a set of images generated according to a variant of the Random Art Algorithm. The Random Art Algorithm, proposed by Andrej Bauer in [**?**], consists of a function RandArt : $x \rightarrow I$ mapping an arbitrary input string x (consisting of the random number generator seed) to an image I, given an set of functions f_i and their probability p_i, $\mathcal{E} = \{(f_1, p_1), \ldots, (f_n, p_n)\}$. The Random Art Algorithm achieves this mapping by generating an expression $\mathcal{F} : [-1 : 1]^2 \rightarrow [-1 : 1]^3$ associating each of an image pixels coordenates, (x, y), to a tone expressed according to the RBG color model, (r, g, b) [3], according to the algorithm RandomArt described in Algorithm [**?**] and considering a $[-1 : 1]$ normalization.

[2] http://www.random-art.org

[3] https://en.wikipedia.org/wiki/RGB_color_model

As a matter of example, consider the set of expressions and respective probabilities $\mathcal{E} = \{(add, 0.2), (sin, 0.2), (perm, 0.2), (rbg, 0.1), ((x, x, x), 0.2), ((y, y, y), 0.1)\}$, where $add(g_1, g_2)$ corresponds to the normalized addiction of any two expressions g_1 and g_2; $sin(g_1)$ corresponds to the sine of any expression g_1; $perm(g_1, g_2, g_3)$ corresponds to a random permutation of any tree expressions g_1, g_2 and g_3; rbg corresponds to a random triple of normalised rgb values. Let the maximum depth of the expression tree be 3. The result, for a seed of $TODO$, is shown in Figure **??** Notice that the algorithm RandomArt will return an expression tree whose leaves are composed of functions whose operatncy is equal to 0, guarranteeing the mapping outputted is indeed $[-1:1]^2 \to [-1:1]^3$.

In order to guarrantee the *Near-one-way Property*, we consider the input string x be the result of a cryptographic hash of an arbitrary input string x'. (we consider the usage of $MD5$ Algorithm [4]). Furthermore, we ensure the *Regularity Property* by computing the Fourier Transform of the Random Art Algorithm output image and asserting whether TOODDDOOO.

This is outlined in algorithm [**?**].

Algorithm 1 Description: Random Art Algorithm Adapted. Returning a image in case it is regular. Otherwise returns False. Notation: Let $f_i^{(p_i, j_i)}$ denote a function of a fixed expression set, where we let p_i denoting the probability of the function being picked from the set and j_i be the operancy (number of input expressions) of the function. Let $\mathcal{E}_1$ be the expression set of function with operancy 0 and $\mathcal{E}_2$ be the expression set of function with operancy greater then 0. Let g denote an expression and x denote a $[-1:1]$ value.

procedure RandomArtMod(x, d)
  ▷ Define Global Variables
  $\mathcal{E}_1 = \{f_i^{(p_i, j_i)} : j_i = 0\}$
  $\mathcal{E}_2 = \{f_i^{(p_i, j_i)} : j_i > 0\}$
  Set ϵ and δ
  ▷ Algorithm Body
  SetSeed($h(x)$)
  $\mathcal{F}$ = RandomArt(d)
  I = DrawImage($\mathcal{F}$)
  if HighFrequencyNoise(I, δ) $< \epsilon$ **then**
  return I
  else
  return False

procedure RandomArt(d)
  if $d \leq 0$ **then**
    f = Random$_{p_1}(\mathcal{E}_1)$ **return** $f(x_1, x_2, x_3)$
  else
    f = Random$_{p_2}(\mathcal{E}_2)$
    for $i = 0; i < j_i; i++$ **do**
      d = Random$_{Unif}([0 : d - 1])$
      g_i = RandomArt$_{\text{Unif}}(d)$
  return $f(g_1, \ldots, g_{j_i})$

[4] https://en.wikipedia.org/wiki/MD5

3.2 System Setup

As mentioned in the section above, we let our Graphical Password alphabet consist of a set of images generated according to a variant of the Random Art Algorithm. Hence, we set up the system by creating a Graphical Password alphabet of size M, where let the input of each of the instances of the Random Art Algorithm be a randomly chosen string. Since we can reproduce each of images being given the input string, we consider only storing the string used to generate each of the images. Notice these string must be save in plain text, which implies that we must assume the system to be secure and trusted, similar to Kerberos [**?**].

3.3 Registration Phase

Upon requesting to registrate, a user is presented with the complete Graphical Password alphabet. Given such alphabet, of size M, the user shall select a password composed of N images. This is show in Figure **??**

3.4 Authentication Phase

Upon requesting authentication, a register user is presented in a challenge. The system regenerates the challenge by randomly selecting a set of n images belonging to the user's portfolio and $n - m$ images not belonging to the user's portfolio, which we identify as *decoy images*. In case any of the images select from the user's portfolio is repeated, the system shall drop one of its examplers and pick an extra image from the set of the decoy images. The images are then randomly placed in the challenge environment.

3.5 Known Attacks

Similarly to what outlined by Dhamija and Perrig in [**?**], we explored a series of possible attacks to the system, and analysed how to protect the system and how to chose its parameters accordingly. For the sake of notation, let Mallory be an attacker who tries to authenticate in the name of a user Alice (impersonation).

1. **Brute-Force Attack.** In this scenario, Mallory attemps to impersonate Alice by successively solving the challenge proposed by the system. Since the challenge consists of selecting an ordered set of n images out of a total m images, the number of possible keys of a single challenge is m^n, yielding a probability of $\frac{1}{m^n}$ of randomly asserting the correct graphical password. Furthermore, notice that since each new challenge randomly selects a new subset of user's password and decoy images from the user's portfolio and from the Graphical Password alphabet, respectively, the probability of randomly asserting the correct graphical password is subsequent drawings does not decay uniformly.

   Table [**?**] provides an overview on the probability of randomly asserting the correct graphical password in a single challenge with respect to the values of m and n.

2. **Observer / Shoulder Surfing Attack.** We identify two possible scenarios in a Observer / Shoulder Surfing Attack. In the first scenario, we consider that Mallory observes and anotes the correct graphical password select by Alice upon completing a challenge. The question of how many challenges Mallory needs to observe in average so to have complete knowledge of Alice's portfolio is addressed by a variant of the well-known *Cuppon Collection Problem* [5], the *Cuppon Collection Problem with constant size groups*, as in [**?**]:

[5] See https://en.wikipedia.org/wiki/Coupon_collector's_problem for details

$$E\{\#\text{trials}\} = \sum_{i=n}^{N} (-1)^{N-i+1} \binom{N}{N-n} \frac{1}{1 - \frac{\binom{i}{n}}{\binom{N}{n}}} + \sum_{i=1}^{n} (-1)^{N-n+i+1} \binom{N}{N-n+i} \quad (1)$$

Table [**?**] provides an overview on the average number of challenges Mallory needs to observe so to have complete knowledge of Alice's portfolio with respect to the values of N and n.

In the second scenario, we consider that Mallory has only knowledge of the placement of the correct graphical password selected within the framework but not of the identity of the image itself. In this case, since the images are randomly placed within the framework, Mallory obtains no information about Alice's portfolio.

3. **Intersection Attack.** In the Intersection Attack, Mallory observes a series of challenges and stores the images in memory. It then tries to recover Alice's portfolio through the information leaked by such knowledge. A simple, but effective procedure, is described in Algorithm [**?**].

Algorithm 2

```
1: procedure IntersectionAttack
2:   W = {1 : 0, ..., M : 0}
3:   for i = 1; i <= num_draws; i++ do
4:     {I^1, ..., I^N} ← Challenge()
5:     for j = 1; j <= n; j++ do
6:       tmp = img2index(I^j)
7:       W[tmp] = W[tmp] + 1
8:   W' = Sort(W)
9:   for i = 1; i <= N; i++ do
10:    I = index2img(W'[i])
11:    if I ∉ P then
12:      return False
13: return True
```

Table **??** provides an overview on the effectiveness of such *Intersection Attack* with respect to the values of m and n, for fixed M and N.

4. **Educated Guess / Social Engineering Attack.**

   In a *Educated Guess / Social Engineering Attack* Mallory tries to take advantage of its knowledge about Alice's image preference and personal taste so to reduce the Graphical Password search space. As mentioned before, the hope is that by using *Random Art* to generate the Graphical Password Alphabet, the randomness associated with the images select by Alice to make up her portfolio, make it increasly difficult to perform such an attack.

3.6 Parameter Choice

In order to choice the system parameters M, N, m and n, we must take into account both the usability of the system and the well-known attacks reviewed in the previous section. The usability of the system concerns the size of the portfolio and ability of a user to easily memorize it and identify it within the challenge. This choice should have been validated thourgh a user study, which was not performed. The protection against the well-known attacks concern

a balance between

Table **??** shows the performance of the parameters chosen against each of the attacks identified.