Cyber and Physical Access Control in Legacy System Using Passwords [1,2]

Securely Manage Passwords with Mobile Phone using Visual Cryptography

Jia XU [a], Jianying ZHOU [a] and Liming LU [b]
[a] *Infocomm Security Department*
Institute for Infocomm Research, Singapore
e-mail: {xuj,jyzhou}@i2r.a-star.edu.sg
[b] *School of Digital Media and Infocomm Technology*
Singapore Polytechnic
e-mail: LU_LIMING@sp.edu.sg

Abstract. Passwords—secret combinations of symbols—play an important role in physical world security (e.g. watchword to prevent unauthorized entry into military forbidden area) from ancient times. With emergence and advance of digital computers and computer network, passwords are also widely adopted in cyber world security protection. In most applications, password protection stands on the frontier of cyber/physical security defense. Compromise of passwords might render the whole system insecure, and make thereafter sophisticated cryptography solution ineffective. However, secure management of a large number of random passwords is a great challenge to human brains. We propose a visual cryptography technique, which allows users to store and manage ciphertexts of randomly chosen passwords in mobile phone and decrypt them *manually* on demand. The stored passwords remain confidential, even if the mobile phone is infected by spyware (Assume the spyware can capture phone screen, and monitor phone CPU and RAM). We also analyze the security and feasibility of proposed method. Leveraging on this technique, we give a simple access control system based on passwords, which provides a low cost alternative solution for legacy system besides smart card based solution.

Keywords. Password Management, Visual Cryptography, Mobile Device, Spyware, Legacy System

1. Introduction

We are interested in finding a low cost (in term of money, training, deploy time and service shutdown time) cyber and physical access control solution to protect cyber and physical assets, for large legacy systems.

[1]This work was supported by the National Research Foundation (NRF), Prime Ministers Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-31) and administered by the National Cybersecurity R&D Directorate.
[2]This work appeared as internal technique report in 2013.

1.1. Legacy System

Some legacy system with long history may still rely on traditional locks (e.g. padlock) to protect each room and important device. For large legacy system, thousands or more lock keys may be required. All management (e.g. distributing keys, labeling keys, searching keys, counting keys, storing keys, backuping keys) of lock keys have to be done manually. More importantly, to revoke a lock key after a staff quits his/her job, the lock and all matching keys have to be replaced by a new pair of lock and keys, which could be expensive.

Modern design of physical door access control widely adopts smart card (e.g. contactless card) based solution. Such smart card based solutions may require network connection and consistent electricity power supply. It might be too expensive for some legacy system to upgrade to modern smart card based solution, since computer network and/or electricity power supply may not reach every corner of some legacy system with large area. In addition, smart card based solution may not be very suitable to protect device or equipment possibly due to network and electricity power requirement.

1.1.1. Access Control using Passwords

We provide a low cost alternative solution for such legacy system: Switch from keyed locks to combination locks, and manage all combination keys, i.e. the passwords (possibly including computer passwords), using staff's own phones, synchronized with a trusted central server. As a result, revocation of a password can be done by a combination lock password reset and a synchronization operation. We remark that, in this solution, the server and user phones communicate via cell phone network (e.g 2G/3G/4G). In case that cell phone network does not cover every inch of the area of legacy system, the user phone can synchronize with the central server where signal is available and then can still manage passwords locally without network connection at any place.

A natural question is that: Is it much easier to steal a password than a lock key? We have to point out that, duplication of a physical lock key may not be essentially more difficult than duplication of a password. One can duplicate a physical lock key just via a photo [5,14], without physically contacting the lock key. With a good phone camera which is widely available nowadays, attacker may even take a photo on a bunch of keys remotely, which could be more serious than peeking passwords over shoulder. Even worse, typically, people are educated to protect password privacy when typing it during login process, but not educated to hide lock keys from others' vision range.

1.2. Password Management

In most (if not all) cyber/physical security system, authentication is the first step, and also one of the most important steps of security protection. Passwords, a secret combination of symbols, is widely adopted in cyber or physical authentication solution, for a long history. Password authentication alone or with other authentication factors (e.g. fingerprint, OTP token), is still the ubiquitous authentication method.

However, the one of most important step—password authentication— is also considered as the weakest link in security protection. It is well known [18] that: (1) a good password should be chosen randomly; (2) passwords should not be reused or shared across different accounts. But, on the other side, (1) random passwords are hard to remember

for human brains; (2) it is even hard, or impossible, to remember many random passwords. Consequently, many users tend to choose simple passwords [19,17], which have a certain pattern and are thus easy to remember, and share the same passwords among different accounts [12]. Unsurprisingly, simple passwords can be easily discovered by brute-force attack with a well-defined dictionary of passwords (called dictionary attack); shared password could be stolen from the account sever with the weakest security protection. Some Internet servers still store users' passwords in plaintext, instead of salted hash. It has witnessed that user password database of some of such servers have been stolen [15,2] by inside or outside attackers. Even if some system mandatorily requires user to setup a complex password, the user might write the complex password on a piece of paper strip and attach the paper strip with his/her computer, where this paper strip is not well-protected.

How to keep and manage multiple passwords securely and conveniently is an interesting and important real-world problem. Due to ubiquitous usage of portable digital device (e.g. smart phone, tablet, PDA) people have developed software (or mobile apps) to manage passwords in digital device. Unfortunately, mobile devices (including Android [4,1,16,3], jailbroken [9]/non-jailbroken [7] iOS [11] and other mobile operating systems) are suffering from stringent threat of spyware, which could log each user input (keystroke or touch points, etc) and even monitor the phone hardware (including screen, RAM, CPU, etc). For an average user, it is hard to tell whether his/her mobile device has been affected by spyware, even with anti-virus software [3].

Based on the above considerations, our goal is to design a simple method that allows users to manage passwords securely and easily in a mobile device, where the mobile device is assumed to be monitored by a spyware.

1.3. Our Contribution

Our main contributions in this work can be summarized below:

1. We propose a simple visual cryptography scheme, which allows users to decrypt a ciphertext *manually*. We also analyze the security of the proposed scheme, against adversary with unbounded computation power.
2. Based on the proposed visual cryptography scheme, we give a simple method to manage passwords in a mobile phone, where the confidentiality of stored passwords retain, even if the phone screen is monitored by possible spyware. Furthermore, we give a method to manage passwords among a large organization and implement access control to cyber or physical resource in large legacy system using passwords.

2. Related Works

2.1. Mobile Password Management

Nowadays, with the wide spread adoption of mobile devices, there is an increasing trend to manage passwords in mobile devices (e.g. smart phone) using a mobile app. Some

[3] Anti-virus's capability is limited, especially for newly emerged malware. In addition, a spyware could disguise itself as an anti-virus app in some loosely controlled app market.

examples found in Google Play Store are: "Keeper Password & Data Vault" [8], "eWallet Password Manager" [6], and "mSecure Password Manager" [13]. However, these password management apps are designed mainly focusing more on convenience than security, and suffer from at least these security issues: (1) Users have to fully trust the password management app itself in privacy of their password: users have no way to prevent the password management app from leaking their passwords to other apps installed in the same device, or from sending their passwords to some server via Internet connection. Even if the source code of password management app is available (i.e. open source program), not every user will invest time or is capable to audit the source code. (2) Still suffer from spyware, since the password will eventually display in the phone screen in some form (e.g. in typed form or handwriting form or fuzzy picture form like CAPCHA [4]), when users want to retrieve the password from the password management app.

In addition, Bojinov *et al.* [24] proposed a method to protect password database on a mobile device from attackers who may have physical access to the mobile device (e.g. stolen phones). Florêncio, Herley and Oorschot studied [27] how to group accounts and passwords for re-use, to achieve balance between security and convenience.

2.2. Visual Cryptography

Visual cryptography, which supports decryption using non-digital mechanical operation, was proposed by Naor and Shamir[29] in 1990's, and after that many subsequent works [26,20,23,28] appear. Most of these works exploit the secret sharing notion proposed by Shamir [30].

3. Our Proposed Visual Cryptography Scheme

In this section, we propose a probabilistic visual encryption scheme (KEYGEN, ENCRYPT, DECRYPT), which allows users to decrypt ciphertext manually. In addition, we also propose an algorithm CIPHERTEXTGEN that generates a random ciphertext. We will analyze the security of the proposed visual cryptography scheme.

3.1. Algorithms Description

In the basic scheme, a plaintext[5] is a string of ℓ symbols from alphabet Σ, a ciphertext is a matrix of `NRow` number of rows and `NCol` number of columns, and an encryption/decryption key is a list of ℓ tuples. All of $\texttt{NRow}, \texttt{NCol}$, and ℓ are public system parameters. A typical setting could be $\texttt{NRow} = 10, \texttt{NCol} = 8, \ell = 8$. In real applications, users could be allowed to choose values for these system parameters. Note that, unlike system parameters `NRow` and `NCol` which are constant across different ciphertexts, choice of alphabet to encrypt each password could be determined by authentication server (what symbols are allowed to form passwords), and different ciphertexts may have different alphabets.

[4] `https://en.wikipedia.org/wiki/CAPTCHA`

[5] Later in Section 4, we will discuss how to support plaintexts of different length.

3.1.1. Key Generation

The probabilistic key generation algorithm takes the system parameter $(\texttt{NRow}, \texttt{NCol}, \ell)$ as input, and outputs an encryption/decryption key K, which is an ordered list of ℓ randomly generated tuples (x_i, y_i, f_i). Here (x_i, y_i) is a coordinate within the grid $[0,\ \texttt{NRow} - 1] \times [0,\ \texttt{NCol} - 1]$, and $f_i \in \{0,1\}$ is a boolean flag. The detailed algorithm is as below.

procedure KEYGEN($\texttt{NRow}, \texttt{NCol}, \ell$)
    Randomly choose ℓ distinct tuple (x_i, y_i)'s from $[0,\ \texttt{NCol} - 1] \times [0,\ \texttt{NCol} - 1]$
    for i from 1 upto ℓ **do**
        $f_i \leftarrow_R \{0,1\}$
    return $K := \{(x_i, y_i, f_i)\}_{i=0}^{\ell-1}$

The generated encryption/decryption key K can be represented *visually* as in Figure 1(a) (on page).

3.1.2. Encryption

The probabilistic encryption algorithm takes an encryption key K, an alphabet Σ, a plaintext $M \in \Sigma^{\ell}$, and system parameters $(\texttt{NRow}, \texttt{NCol}, \ell)$ as input. The output (i.e. the ciphertext) of the encryption algorithm is a matrix T of dimension NRow by NCol, where each cell is filled with a symbol from the alphabet Σ.

procedure ENCRYPT($K, \Sigma, M, \texttt{NRow}, \texttt{NCol}, \ell$)
    Create an empty matrix T with dimension NRow by NCol
    for i from 0 upto $\ell - 1$ **do**
        Parse $K[i]$ as (x_i, y_i, f_i)
        $m_i \leftarrow M[i]$
        if f_i equals 1 and m_i is a letter **then**
            $m_i \leftarrow \texttt{toggleUpperLowerCase}(m_i)$
        $T[x_i][y_i] \leftarrow m_i$
    $q \leftarrow \texttt{floor}(\texttt{NRow} \times \texttt{NCol} / |\Sigma|)$
    $r \leftarrow \texttt{NRow} \times \texttt{NCol}$ **modulo** $|\Sigma|$
    Randomly choose a subset S of size r from Σ
    for i from 1 upto q **do**
        $\Sigma_i \leftarrow \Sigma$
    Multiset $W \leftarrow \Sigma_1 \cup \Sigma_2 \cup \ldots \Sigma_q \cup S$
    Initiate set $Z \leftarrow \emptyset$
    for each symbol a in M **do**
        if $a \in W$ **then**
            $W \leftarrow W \setminus \{a\}$ `/* Remove a from multiset W */`
        else
            $Z \leftarrow Z \cup \{a\}$
    Randomly choose $|Z|$ elements from W and replace them by Z
    Fill all elements in multiset W to all empty cells of T
    Randomly permutate all cells in T, excepted locations specified by key K
    return T

In our application, the encryption algorithm runs in a digital computing device, e.g. mobile phone. An example of ciphertext is visually represented in Figure 1(b).

3.1.3. Decryption

The deterministic decryption algorithm takes a decryption key K, a ciphertext T, and system parameters $\texttt{NRow}, \texttt{NCol}, \ell$ as input. The output is a string of ℓ symbols from the alphabet Σ, where Σ is specified in encryption process.

procedure DECRYPT($K, T, \texttt{NRow}, \texttt{NCol}, \ell$)
  for i from 0 upto $\ell - 1$ **do**
    Parse $K[i]$ as (x_i, y_i, f_i)
    $m_i \leftarrow T[x_i][y_i]$
    if f_i equals 1 and m_i is a letter **then**
      $m_i \leftarrow \texttt{toggleUpperLowerCase}(m_i)$
  return $m_0 m_1 \ldots m_{\ell-1}$

This decryption process is just to locate cells in the ciphertext matrix specified by decryption key, retrieve symbols in these cells, and flip the case of some letter symbol if condition meets. Such a simple procedure can be done mechanically or even manually, without any digital computing device. Figure 1(c) illustrate how to perform this decryption operation manually.

3.1.4. Random Ciphertext Generation

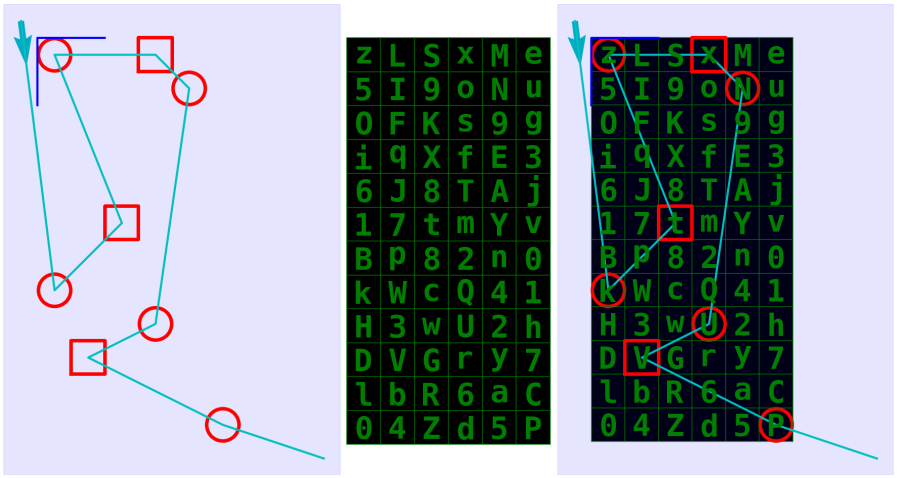
In addition, we also propose an extra algorithm CIPHERTEXTGEN, which takes as input an alphabet Σ and system parameters $\texttt{NRow}, \texttt{NCol}$, and outputs a random matrix T with cells filled by symbols from alphabet Σ.

procedure CIPHERTEXTGEN($\Sigma, \texttt{NRow}, \texttt{NCol}$)
  $q \leftarrow \texttt{floor}(\texttt{NRow} \times \texttt{NCol} / |\Sigma|)$
  $r \leftarrow \texttt{NRow} \times \texttt{NCol} \mod |\Sigma|$
  if q equals 0 **then**
    Abort
  Randomly choose a subset S of size r from Σ
  for i from 1 upto q **do**
    $\Sigma_i \leftarrow \Sigma$
  Multiset $W \leftarrow \Sigma_1 \cup \Sigma_2 \cup \ldots \Sigma_q \cup S$
  Create an empty matrix T with dimension $\texttt{NRow}$ by $\texttt{NCol}$
  Fill all elements of multiset W into cells of matrix T
  Random permutate all cells of matrix T
  return T

3.1.5. How to Save the Decryption Key

Users may make a physical decryption key token by DIY. Here we give two examples: (1) Draw the key shown in Figure 1(a) on a piece of hard transparent plastic card. (2) Dig square or circle holes according to the key on a piece of hard paper (e.g. a name card), and draw a directed curve to connect all holes. To decrypt a ciphertext matrix displayed in a phone screen, users just put the physical decryption key token over the phone screen and align them well, then users can figure out the hidden password in mind quickly.

The unit cost of a key token is low, and can be further reduced if a lot of such physical tokens are produced in factory. The advantage of such physical decryption key



(a) Encryption/Decryption Key. (b) Ciphertext (c) Visual Decryption

Figure 1. Illustration of Visual Decryption: : Align the blue color right-angle at the left-top corner of the decryption key in Figure 1(a) with the left-top corner of the ciphertext matrix in Figure 1(b), and read out message "kTzXNUvP" in Figure 1(c), where the case of letters selected by square is toggled and unchanged if selected by circle.

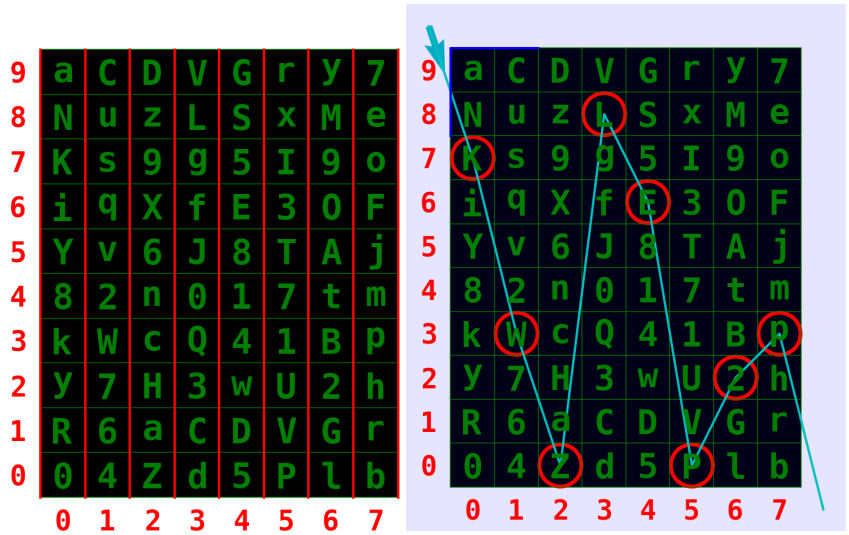
token is that users require to do a little mental effort. The drawback is that it could be lost or stolen.

3.1.6. How to Memorize the Decryption Key

Alternatively, users could remember the decryption key in their brains. Here we provide some suggestions to help users memorize the decryption keys. The advantage of this option is that no extra physical token is required and thus more convenient, but the entropy of the decryption key (arguably) might reduce, and we should make sure the remaining entropy will be still sufficient to protect passwords.

In order to easily remember the decryption key in human brain, one can choose the decryption key with a certain pattern. Some examples are as below:

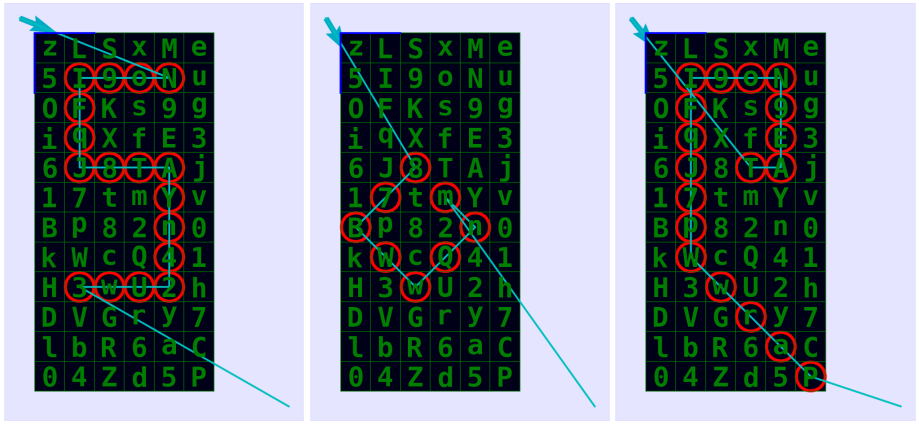
1. Encode key as numbers: Let $\texttt{NRow} = 10$, i.e. the number of rows in ciphertext matrix is 10. We can encode a subclass of decryption keys with digits. Any ℓ digital number $x_0, x_1, \ldots, x_i, \ldots, x_{\ell-1}$, each x_i in the range $[0,9]$, encode a decryption key $(0,x_0),(1,x_1),\ldots,(i \mod \texttt{NCol}, x_i,), \ldots, (\ell-1 \mod \texttt{NCol}, x_{\ell-1})$, ordered from left to right, where all $f_i = 0$. An example is showed in Figure 2.
2. Graphical Pattern: We may conceptually walk inside the ciphertext matrix according to some graphical pattern. For example, Figure 3(a) shows shape of digital "5"; Figure 3(b) shows shape of a diamond; Figure 3(c) shows shape of a flip version of digit "9". Similarly, one can also walk in the ciphertext matrix according to an English letter, or a character from any other language, or any shape he/she likes. To remember such decryption key, the users need only remember some anchor point positions. Note that in Figure 3, the key length may be very large. Users may take a substring of the long key as the actual decryption key. E.g. (1) substring of symbols at odd positions; (2)Deterministically derive a small



(a) A Ciphertext

(b) Visual representation of key encoded by digits "73086023"

Figure 2. Example of decryption key that can encodes to numbers. The labels along X-axis and Y-axis will help users to locate cells manually. Thick vertical lines in Figure 2(a) will mark columns obvious.



(a) 5

(b) Diamond

(c) Flip 9

Figure 3. Example of Decryption Keys with Simple Graphic Pattern

integer v, say $v \in [0,4]$, from the ciphertext label (or auxiliary) information, and let the suffix starting at position v of the long key be the actual decryption key.

It is worthy to point out that, every time the user wants to retrieve any plaintext (e.g. passwords), he/she should recall his/her decryption key mentally, in order to perform the decryption operation manually. The more frequent exercises of such manual decryption, the better that users could memorize the decryption key. Some previous works [25,21,22] showed that it is possible to remember a high entropy secret by a human brain, with a

certain method (e.g. rehearsing the passwords in proper time period).

3.2. Security Analysis of Proposed Visual Cryptography Scheme

Let $\mathbf{H}(\cdot)$ denote Shannon's entropy function, and $\mathbf{H}(\cdot|\cdot)$ denote the conditional entropy. Let $\mathbf{K}, \mathbf{M}, \mathbf{T}$ denote the random variables for key, plaintext, and ciphertext, respectively. In this subsection, if not otherwise specified, we assume the encryption/decryption key $\mathbf{K}$ is chosen by probabilistic algorithm KEYGEN, the ciphertext $\mathbf{T}$ is chosen by probabilistic algorithm CIPHERTEXTGEN, and the plaintext $\mathbf{M}$ is derived deterministically from decryption key $\mathbf{K}$ and ciphertext $\mathbf{T}$ using deterministic algorithm DECRYPT. This is very different from traditional security setting of encryption scheme, where plaintext is chosen by user according to some distribution and ciphertext is (probabilistically) derived from encryption key and plaintext. The root cause of the difference is that, our encryption scheme will be applied to a special message — password, which is supposed to be random without any semantics.

We will quantify the entropy of decryption key $\mathbf{K}$ in various attack mode. High entropy decryption key $\mathbf{K}$ and random ciphertext $\mathbf{T}$, will guarantee that the message $\mathbf{M}$ (i.e. password to be protected) will have high entropy and thus secure.

Lemma 1. $\mathbf{H}(\mathbf{K}) \geq \ell \log \ell + \log \binom{\texttt{NRow}\times\texttt{NCol}}{\ell}$.

Proof. An encryption/decryption key consists of an ordered list of ℓ randomly chosen distinct coordinate pairs (x_i, y_i) in the grid of size NRow by NCol, and ℓ random boolean flag values f_i, which indicates user to toggle the upper/lower case of letter symbol.

The number of possible ordered lists of ℓ distinct coordinate pairs is $\binom{\texttt{NRow}\times\texttt{NCol}}{\ell} \times \ell!$. Thus, the entropy of key $\mathbf{K}$ is $\mathbf{H}(\mathbf{K}) \geq \log\left(\binom{\texttt{NRow}\times\texttt{NCol}}{\ell} \times \ell!\right) \geq \ell \log \ell + \log \binom{\texttt{NRow}\times\texttt{NCol}}{\ell}$.

Notice that, the number of possible lists of ℓ boolean values f_i's is 2^ℓ, which may contribute additional ℓ bits entropy to $\mathbf{H}(\mathbf{K})$ when every symbol (e.g. English letter) in the alphabet has upper case and lower case. However, when some or even all symbols in the alphabet has only one form (e.g. digits) without distinction of upper/lower case, such contribution of boolean values f_i's to $\mathbf{H}(\mathbf{K})$ could be some real value in the range $[0, \ell]$. □

Lemma 2 (Ciphertext-Only Attack I)**.** $\mathbf{H}(\mathbf{K} \mid \{\mathbf{T}_i\}_i) = \mathbf{H}(\mathbf{K})$.

Proof. This lemma holds, since the key $\mathbf{K}$ and ciphertexts $\mathbf{T}_i$'s are independently chosen. □

Lemma 3 (Ciphertext-Only Attack II)**.**
$\mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n \mid \{\mathbf{T}_i\}_{i=1}^n) = \mathbf{H}(\mathbf{K})$*, where* $\mathbf{M}_i = \text{DECRYPT}(\mathbf{K}, \mathbf{T}_i, \texttt{NRow}, \texttt{NCol}, \ell)$

Proof. Recall a generic rule of Shannon's entropy: $\mathbf{H}(X,Y) \geq \mathbf{H}(X)$, for any random variable X and Y. Using this rule together with the definition of conditional entropy, it is easy to derive:

$$\mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n \mid \{\mathbf{T}_i\}_{i=1}^n) \tag{1}$$

$$\geq \mathbf{H}(\mathbf{K} \mid \{\mathbf{T}_i\}_{i=1}^n) \tag{2}$$

$$= \mathbf{H}(\mathbf{K}) \qquad \text{(by Lemma 2)}. \tag{3}$$

On the other side, we have

$$\mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n \mid \{\mathbf{T}_i\}_{i=1}^n) + \mathbf{H}(\{\mathbf{T}_i\}_{i=1}^n) \tag{4}$$

$$= \mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n,\ \{\mathbf{T}_i\}_{i=1}^n) \tag{5}$$

$$= \mathbf{H}(\mathbf{K},\ \{\mathbf{T}_i\}_{i=1}^n) \tag{6}$$

$$\leq \mathbf{H}(\mathbf{K}) \ +\ \mathbf{H}(\{\mathbf{T}_i\}_{i=1}^n) \tag{7}$$

$$\implies \quad \mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n \mid \{\mathbf{T}_i\}_{i=1}^n) \leq \mathbf{H}(\mathbf{K}). \tag{8}$$

Note that Eq (5) equals to Eq (6), is because that messages $\mathbf{M}_i$'s are deterministic function values (i.e. DECRYPT) of key $\mathbf{K}$ and ciphertexts $\mathbf{T}_i$'s.

Combining both upper and lower bounds, we have $\mathbf{H}(\mathbf{K},\ \{\mathbf{M}_i\}_{i=1}^n \mid \{\mathbf{T}_i\}_{i=1}^n) = \mathbf{H}(\mathbf{K})$. □

Lemma 4 (Known-Plaintext Attack)**.** *Suppose* $\texttt{NRow} \times \texttt{NCol} \geq |\Sigma| \times q$ *for some integer* q*. Suppose* $\mathbf{M} \in \Sigma^\ell$ *consists of* n_j *number of symbols* $s_j \in \Sigma$*,* $j \in [0, v]$*, where* $\sum_{j=0}^{v} n_j = \ell$ *and all symbols* s_j*'s are distinct. We have* $\mathbf{H}(\mathbf{K} \mid (\mathbf{T}, \mathbf{M})) \geq \log\left(\prod_{j=0}^{v} \binom{q}{n_j} \times n_j!\right)$.

Particularly, if plaintext $\mathbf{M}$ *consists of* ℓ *distinct symbols, we have* $\mathbf{H}(\mathbf{K} \mid (\mathbf{T}, \mathbf{M})) \geq \ell \log q$*. Additionally, if the plaintext* $\mathbf{M}$ *consists of* ℓ *distinct English letters, then* $\mathbf{H}(\mathbf{K} \mid (\mathbf{T}, \mathbf{M})) \geq \ell \log 2q$.

If the decryption key is encoded as ℓ numbers in $[0, 9]$ and $\texttt{NRow} = 10$, the number of possible combinations for decryption keys is 10^ℓ , which could be sufficient (e.g. $\ell \geq 8$) to defend brute force attack, if some delay mechanism is adopted in login process, although it is smaller than the total number of all possible decryption keys. For graphical pattern key, the key space could be even larger. Theoretical and empirical analysis of space size of graphical pattern key can be found in existing works [31],[32].

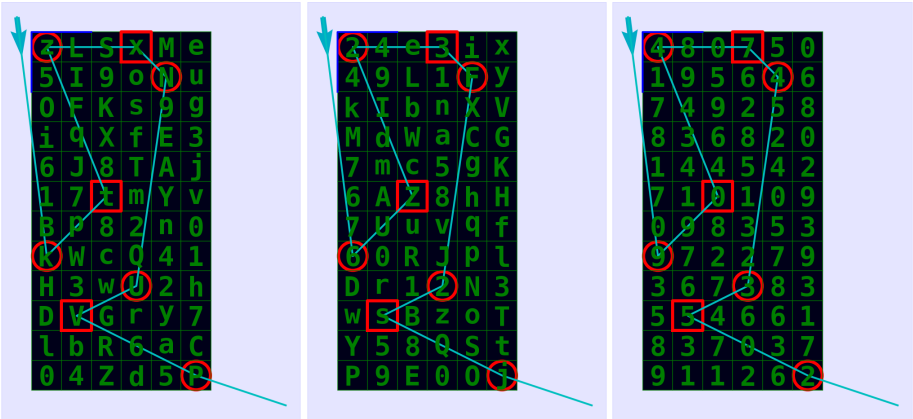
In summary, if the plaintext is uniformly random distributed over its alphabet set, and the adversary can only do a limited amount (e.g. smaller than 10^6) of brute force search, then our proposed visual cryptography scheme is secure. We remark that if the plaintext is not uniformly distributed, the proposed visual cryptography scheme could be broken by frequency analysis, similar like traditional substitution cipher.

4. Managing Passwords using Visual Cryptography

In previous section, we proposed a secure visual cryptography scheme for uniformly random plaintexts. An ideal application of this scheme is to encrypt passwords: because that (1) a secure password has to be randomly generated; (2) typically, adversary is not allowed [6] to do a large number of brute force attack (e.g. passwords with 6 digits of $\log 10^6$ bits entropy could be sufficiently secure for some authentication system).

Let Σ be the alphabet for passwords, and a password is a string of ℓ symbols from Σ. We generate an encryption/decryption key and apply the encryption algorithm ENCRYPT described in previous section, to encrypt the password, and store the ciphertext in a mo-

[6] Many authentication systems have mechanisms to essentially slow down online or even offline brute-force attack on passwords.



(a) “kTzXNUvP” (b) “6z23F2Sj” (c) “90474352”

Figure 4. Illustration of visual decryption with one key and three distinct ciphertexts, where in the first two ciphertexts, the alphabet consists of English letters and digits, and in the third ciphertext, the alphabet consists of only digits. Each sub-figure is labeled with the corresponding decrypted message.

bile phone or print it on a piece of paper. Alternatively, we could generate a random ciphertext using algorithm CIPHERTEXTGEN and then obtain a password by decrypting the ciphertext with decryption key using algorithm DECRYPT.

4.1. Visually Encrypt Passwords

4.1.1. Short Passwords

In case that the length of a password s is smaller than ℓ, we can append a random padding suffix (randomly chosen from $\Sigma^{\ell-|s|}$) to the password, so that the resulting string has length ℓ. Then we can apply the above method to encrypt this padded password to obtain a random matrix T as ciphertext. Meanwhile the length $|s|$ of real password is recorded in plaintext. During decryption, the user just ignore the last $(\ell-|s|)$ symbols in the output of DECRYPT algorithm.

4.1.2. Long Passwords

If the length of a password s is larger than ℓ, we can divide it into blocks of length ℓ, where possibly the last (partial) block may be expanded to a full bock using random padding. Then the user can encrypt each block of length ℓ using the proposed visual encryption scheme, to get a random matrix for each block. Similar as in previous case of short password, the length $|s|$ of actual password will be recorded in plaintext.

4.1.3. Multiple Passwords

We can encrypt all passwords separately using the same encryption key. Each ciphertext will be labeled with the account information (e.g. room number, device id, or domain name and id), to distinguish with each other. Figure 4 shows that one key encrypts three passwords with different alphabets.

4.1.4. New Passwords Generation

To generate a new password from alphabet Σ with length ℓ, we simply generate a random ciphertext by revoking CIPHERTEXTGEN$(\Sigma, \texttt{NRow}, \texttt{NCol})$, and visually decrypt it using the decryption algorithm DECRYPT and decryption key. The decrypted message will be the password from space Σ^{ℓ}. Shorter or longer passwords can be generated using similar ideas as above.

In case that the authentication system has more restriction on passwords, e.g. every password has to contain digits, upper case letters and lower case letters, we can keep trying to generate new random passwords, until we find one that meets the requirement.

There are two approaches to generate and save a new password: (1) Generate a password using some other method and then encrypt it by running ENCRYPT algorithm using a mobile phone app, then save the ciphertext. (2) Run CIPHERTEXTGEN algorithm to generate a ciphertext using a mobile phone app, and then *manually* and *mentally* decrypt the ciphertext to get the new password on the demand. The second method will be more secure in the sense that the mobile phone app is fully ignorant to the new password, while in the first method, the mobile app has full knowledge of the new password. Note that it is not convenient to perform our encryption method manually.

4.2. Management Passwords on a Mobile Phone

We implement a mobile app in android phone, to execute our visual cryptography scheme and manage passwords. Any user can manage his/her passwords in digital devices by following the below steps:

1. User chooses and remembers a master password to lock the mobile app and all ciphertexts (this master password can be replaced by fingerprint, if fingerprint authentication is supported in the device. Such master password or fingerprint will protect users' stored passwords, in case that both the mobile device and the physical decryption key token are stolen by attackers).
2. User randomly chooses an encryption/decryption key and keeps it private.
3. For each account, User generates a new random password by running the CIPHERTEXTGEN algorithm with the mobile app and label the newly generated ciphertext with the account information. Then User re-sets password for this account in corresponding authentication system.
4. To retrieve a designated password for a particular account,
   (a) User unlocks the mobile app by providing the master password (or fingerprint);
   (b) User searches the corresponding ciphertext in the digital device;
   (c) Decrypt the ciphertext to get password manually.
   (d) User manually types the password to the login interface on the login device, where the login device is supposed to be different from the device that stores and manages passwords.
   About Step (c) and (d), we emphasize that, decrypting a ciphertext of password and typing a password into the login device could be in parallel and in one symbol by one symbol manner. That is, a user could manually decrypt the first symbol of password and type this symbol into the login device, and then proceed for the second symbol, and then for other symbols one by one.

5. Lock the mobile app and remove all unprotected version of ciphertexts from the digital device memory.

We emphasize that, very different from Google's Android graphical screen lock, in our method, **users are not supposed to touch symbols** in the ciphertext matrix, which is displayed **in the phone screen**.

4.2.1. Scope of Applications

We recommend that our method should be used in the following way:

- (**Condition 1**) the login device is different from the mobile device which manages passwords using our method. E.g. passwords of bank ATM card, safe box, combination lock, computer server root password, etc.
- (**Condition 2**) authentication servers of all passwords to be managed should have a comparable level of security protection. For example, ordinary website (e.g. some online forum, Facebook, twitter) has arguably weaker security protection on users' password than large bank corporation.

The reason behind condition 1 is that: If a mobile phone is affected by spyware, and our method is used to manage passwords of website (e.g. Facebook), the password could be captured by spyware, when a user types the password to the website login interface using the same mobile phone, although storage of passwords is secure.

The reason behind condition 2 is that: Our method has a certain level (but still limited) resistance to known-plaintext attack, if multiple plaintext-ciphertext pairs are given. Some websites may still store all users' passwords in plaintext in server side for some reason, and these passwords might be stolen by hackers and revealed in Internet. If the spyware in mobile phone has access to such leaked password database, it might be able to launch known-plaintext attack on our scheme. In physical world, no centralized authentication server is required by combination lock, which is favorite to our scheme.

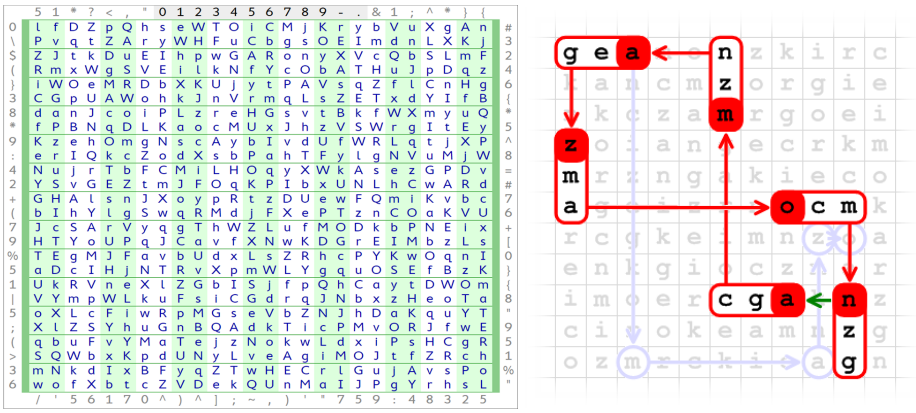
We recommend users to classify all passwords in categories, according to the security levels of authentication servers and value of assets protected by each password, and protect each category of passwords using a separate visual cryptography key. For example, choose a random visual cryptography key K_0 to manage passwords of high value, e.g. ATM passwords, safe-box passwords, computer server root passwords, using our proposed method, and never use the master key K_0 to protect passwords of normal websites (e.g. Facebook, twitter, online forum).

4.2.2. Comparison

It is easy to see that, our mobile password management solution based on our visual cryptography method, is more secure than existing mobile password management apps (e.g. [8,6,13]), in the sense that, our solution can defeat spyware, and in other solutions, spyware can obtain all protected passwords by monitoring the phone screen. In any other security aspect, our solution is at least equally good as others.

4.3. Our Experiment

We implement our method in android phone and conduct survey with 7 subjects (teenage students). With little training (time for briefing and training is within 5 minutes), it takes



(a) The 26 by 26 Latin Square used in Off The Grid method

(b) Illustration in a smaller 11 by 11 Grid: Domain name "amazon" enciphers to password "gcznegmacmzg"

Figure 5. Off The Grid. Both pictures are from `https://www.grc.com/offthegrid.htm`

45.3 seconds on average for these 7 subjects to successfully login to a randomly chosen online account among a list of 100, using our mobile app, where decryption key of our visual encryption scheme is remembered in subject's brain. Here searching time, manual decryption time and login time are all included. The shortest time record is 30 seconds and the longest one is 64 seconds. In the future, we will also experiment on opening a combination lock.

4.4. Alternative Visual Cryptography for Password Management

4.4.1. Off The Grid

Steve Gibson proposed a paper based system (called as "Off The Grid") for encrypting[7] a domain name into a random looking password using Latin Square [10], where the resulting password can be used for this domain exclusively. In this method, Latin Square (see Figure 5(a)) is a 26 by 26 grid and filled in with $26 \times 26 = 676$ upper or lower case English letters, in a way such that, each English letter (ignore the case) will appear *exactly* once in each row and in each column. A Latin Square will be generated randomly using a computer program and printed on a piece of paper. Given a starting point, a domain name will determine a unique path in a Latin Square. The encryption process (see Figure 5(b)) is to walk along this unique path in the Latin Square and retrieve neighboring letters manually. To support special symbols (e.g. digits, dots, and hyphen) rather than English letters in the input/output, the author has to patch this scheme, and thus require certainly more complex procedures to do manual decryption, compared with our method.

4.4.2. Comparison with Our Method

In this "Off The Grid" method, the paper with printed Latin Square plays the role of secret key and should be kept securely, the domain names (after possible substitution from

[7]The author describes it as "encryption". But according to the algorithm, it is more like a hash instead of encryption method, since decryption algorithm is not required, and not provided either.

Table 1. Comparison between Our Scheme and Off The Grid Method

Our Scheme	**Off The Grid**
Probabilistic encryption. The ciphertext is a matrix of symbols.	Deterministic encryption (actually hash). The key (i.e. Latin Square) is a matrix of symbols.
Users manually decrypt a ciphertext to obtain password	Users manually encrypt/hash an input message to obtain password
Account id is labeled with the ciphertext of password	Account id is the input message to the encrypt/hash method
Dimension is flexible (Suitable for both phone screen or printed on paper)	26 by 26 Grid (Difficult to see if shown in phone screen for manual encrypting/hashing)
Encryption key and alphabet are independent (Example of one key, three ciphertexts and two distinct alphabets is given in Figure 4)	Encryption key binds to a fixed alphabet. Additional patches required to support digits and special symbols in input and/or output (See the digits and special symbols around the boarders of the Latin Square in Figure 5(a)). The resulting output is a repeation of pattern "letter letter non-letter" (`https://www.grc.com/otg/enhancements.htm`)
Passwords remain secure if matrix of symbols is leaked	All passwords revealed if Latin Square is leaked

non-alphabet symbols to alphabet symbols) are plaintexts, and the resulting passwords are ciphertexts. We compare this method with our method in Table 1

5. Access Control in Legacy System using Passwords

In cyber access control and authentication system, passwords are widely used to protect each computer, server and online resource. Especially, legacy systems may rely more on passwords for cyber access control, compared with newly designed system which may prefer smart card and biometric authentication (e.g. fingerprint and iris). For physical access control, some legacy system may still reply on keyed lock (e.g. padlock) to protect each room and important device. We suggest such legacy system to switch from keyed lock to combination lock, and securely and conveniently manage all passwords, for both cyber and physical access control, using our visual cryptography method in mobile phones.

5.1. Current access control in Legacy system

In some legacy system, keyed lock is widely adopted to protect access to rooms or important devices. Some staff has to carry a lot of keys during work time, and find a correct one to open a lock to a door or device. As a physical token, keys have some inherent disadvantages: (1) it is relatively expensive to revoke or update a key; (2) all management (e.g. distributing/labeling/searching/storing/backuping keys) of keys are performed manually and are thus troublesome; (3) it is troublesome for a staff to carry a lot of keys. Some organization might share the same key for different rooms/devices, in order to reduce the number of keys and simplify key management, at the cost of security. For a large legacy system, the number of keys could be thousands or more.

5.2. Upgrade to Password based System

In newly designed modern systems, smart card (e.g. contactless card) based door solution is widely adopted.

Smart card solution requires persistent electronic power supply and networks. This could be the reason that it is only widely applied to protect rooms, instead of devices. Some legacy system may spread in a large area, and their existing power supply grid and computer network may not cover every room to be protected. Despite the great security and convenience that smart card based door access solution offers, the upgrading cost might be too high for such legacy system with the traditional keyed locks, considering the overall cost including power grid and network construction.

For such legacy system, we propose an alternative mitigation solution: (1) switch keyed-lock to password based combination lock (e.g. single dial combination lock); (2) end users store and manage passwords on their own mobile phones using our visual encryption method; (3) build a trusted central server, to distribute, synchronized, update, revoke all passwords, among all users.

The trusted central server could communicate with users mobile phones in a secure communication channel using cell phone network (e.g. 2G(SMS)/3G/4G), or wifi in a small area. When users move to location beyond the coverage of cell phone network of wifi, users can manage passwords locally without network connection.

The total cost of our solution will be limited: (1) Optionally, users bring their own phone to work, which means zero additional cost for end device; (2) In case that cell phone network available, just pay the additional communication data usage and no additional cost to construct the network facility; in case of wifi, the cost of wifi router is cheap; (3) Setup a central server.

6. Conclusion

We proposed a visual cryptography scheme to encrypt random messages and applied it to protect and manage passwords in digital devices. Our method protects confidentiality of passwords even if in an extreme case that the digital device is monitored by spyware. In other words, our method can protect the confidentiality of passwords, without trusting the digital device and without trusting the password management app which implements our method. Our method provides two alternative configurations: (1) higher security with a dedicated non-digital physical token (e.g. made with a piece of paper or plastic card); (2) sufficient security without any extra physical token. Our method protects the storage confidentiality of passwords, and can increase the overall security level, if the login device is different from the digital device that stores ciphertexts of passwords. Thus, our method is very suitable to manage combination lock passwords, safe box passwords, ATM passwords, computer login password, server root password, and so on. Furthermore, our method is easy to understand (so easy to audit security of our proposal) and easy to use, and the deployment cost is cheap.

In future work, we may design more sophisticated physical decryption token (say optical based) to provide a much larger key space.

References

[1] 16 million mobile devices infected with malware, http://www.esecurityplanet.com/mobile-security/16-million-mobile-devices-infected-with-malware.html

[2] 42 million unencrypted passwords leaked, http://www.computerworld.com/article/2475534/cybercrime-hacking/42-million-unencrypted-passwords-leaked-from-hacked-online-dating-site-cupid-medi.html

[3] android malware ios-apps pose greater risk leaking user data, http://tech.firstpost.com/news-analysis/android-malware-ios-apps-pose-greater-risk-leaking-user-data-says-study-219746.html

[4] android malware rises 300 percent, http://www.scmagazine.com/android-malware-rises-300-percent-report-says/article/390903/

[5] Copy your house keys with your phone, https://keysduplicated.com/

[6] eWallet Password Manager, https://play.google.com/store/apps/details?id=org.awallet.free&hl=en

[7] ios devices dont have to be jailbroken for spyware, http://www.scmagazine.com/ios-devices-dont-have-to-be-jailbroken-for-spyware-sold-by-hacking-team-to-be-installed/article/426137/

[8] Keeper Password and Data Vault, https://play.google.com/store/apps/details?id=com.callpod.android_apps.keeper&hl=en

[9] keyraider malware stole over 225000 apple credentials from jailbroken ios devices, http://www.computerworld.com/article/2978152/cybercrime-hacking/keyraider-malware-stole-over-225-000-apple-credentials-from-jailbroken-ios-devices.html

[10] Latin Square, http://mathworld.wolfram.com/LatinSquare.html

[11] Malware for iOS, https://www.theiphonewiki.com/wiki/Malware_for_iOS

[12] Measuring password re-use empirically, https://www.lightbluetouchpaper.org/2011/02/09/measuring-password-re-use-empirically/

[13] mSecure Password Manager, https://play.google.com/store/apps/details?id=com.mseven.msecure&hl=en

[14] Software can clone keys from single photo, http://www.telegraph.co.uk/news/newstopics/howaboutthat/3375872/Software-can-clone-keys-from-single-photo.html

[15] sony hacked yet again plaintext passwords posted, http://arstechnica.com/tech-policy/2011/06/sony-hacked-yet-again-plaintext-passwords-posted/

[16] spyware pre-installed on certain xiaomi, huawei and lenovo smartphones sold by third party, http://www.digit.in/mobile-phones/spyware-pre-installed-on-certain-xiaomi-huawei-and-lenovo-smartphones-27144.html

[17] the 25 most popular passwords of 2014, http://gizmodo.com/the-25-most-popular-passwords-of-2014-were-all-doomed-1680596951

[18] US-Cyber Emergency Response Readiness Team: CyberSecurity Tips. (Choosing and Protecting Passwords), http://www.us-cert.gov/cas/tips/

[19] worst passwords 2013, http://splashdata.com/press/worstpasswords2013.htm

[20] Ateniese, G., Blundo, C., De Santis, A., Stinson, D.R.: Visual cryptography for general access structures. Information and Computation 129(2), 86–106 (Sep 1996)

[21] Blocki, J., Blum, M., Datta, A.: Naturally Rehearsing Passwords. CoRR abs/1302.5122 (2013), http://arxiv.org/abs/1302.5122

[22] Blocki, J., Komanduri, S., Cranor, L.F., Datta, A.: Spaced Repetition and Mnemonics Enable Recall of Multiple Strong Passwords. CoRR abs/1410.1490 (2014), http://arxiv.org/abs/1410.1490

[23] Blundo, C., De Santis, A., Naor, M.: Visual Cryptography for Grey Level Images. Information Processing Letters 75(6), 255–259 (Nov 2000)

[24] Bojinov, H., Bursztein, E., Boyen, X., Boneh, D.: Kamouflage: Loss-resistant Password Management. In: Proceedings of the 15th European Conference on Research in Computer Security. pp. 286–302. ESORICS'10 (2010)

[25] Bonneau, J., Schechter, S.: Towards Reliable Storage of 56-bit Secrets in Human Memory. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 607–623 (2014)

[26] Droste, S.: New Results on Visual Cryptography. In: CRYPTO 96: Advances in Cryptology. pp. 401–415

[27] Florêncio, D., Herley, C., van Oorschot, P.C.: Password Portfolios and the Finite-Effort User: Sustainably Managing Large Numbers of Accounts. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 575–590 (2014)
[28] Hou, Y.C.: Visual cryptography for color images. Pattern Recognition 36(7), 1619–1629 (2003)
[29] Naor, M., Shamir, A.: Visual Cryptography. In: EUROCRYPT '94: Advances in Cryptology. pp. 1–12
[30] Shamir, A.: How to Share a Secret. Commun. ACM 22(11), 612–613 (Nov 1979)
[31] Tao, H., Adams, C.: Pass-Go: A Proposal to Improve the Usability of Graphical Passwords 7(2), 273–292 (Sep 2008)
[32] Uellenbeck, S., Dürmuth, M., Wolf, C., Holz, T.: Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer; Communications Security. pp. 161–172. CCS '13 (2013)