Abstract

A fundamental task in security is to create cryptographic primitives based on hard mathematical problems that are computationally intractable. Using hard AI (Artificial Intelligence) problems for security is an exciting new paradigm. Under this paradigm, the most notable primitive invented is Captcha, which distinguishes human users from computers by presenting a challenge, i.e., a puzzle, beyond the capability of computers but easy for humans. To create new security primitive based on hard AI problems is a challenging and interesting open problem. Here new idea is introduced, a new security primitive based on hard AI problems, namely, a novel family of graphical password systems integrating Captcha technology, which is called CaRp (Captcha as graphical Passwords). CaRp is click-based graphical passwords, where a sequence of clicks on an image is used to derive a password. Unlike other click-based graphical passwords, images used in CaRp are Captcha challenges.

1. Introduction

1.1 Why Graphical Passwords?
Access to computer systems is most often based on the use of alphanumeric passwords. Though, users have difficulty remembering a password that is long and random-appearing. Instead, they create short, simple, and insecure passwords. Using a graphical password, users click on images rather than type...
alphanumeric characters. Graphical passwords have been designed to try to make passwords more memorable and easier for people to use and, therefore, more secure.

**CAPTCHA- Completely Automated Public Turing test to tell Computers and Humans Apart**

---

### 1.2 Captcha As Graphical Passwords (CaRP)

A typical way to apply CaRP schemes in explained in Flowchart 1.2. The authentication server AS stores a salt $s$ and a hash value $H(p, s)$ for each user ID, where $p$ is the password of the account and not stored. A CaRP password is a sequence of visual object IDs or clickable-points of visual objects that the user selects. Upon receiving a login request, AS generates a CaRP image, records the locations of the objects in the image, and sends the image to the user to click her password. The coordinates of the clicked points are recorded and sent to AS along with the user ID. AS maps the received coordinates onto the CaRP image, and recovers a sequence of visual object IDs or clickable points of visual objects, that the user clicked on the image. Then AS retrieves salt $s$ of the account, calculates the hash value of with the salt, and compares the result with the hash value stored for the account. Authentication succeeds only if the two hash values match.

![Figure 1.1: Some CAPTCHA Styles](image)

**Figure 1.1: Some CAPTCHA Styles**

![Flowchart of Basic CaRP Authentication](image)

**Figure 1.2: Flowchart of Basic CaRP Authentication**

**Flowchart of Basic CaRP Authentication**

<table>
<thead>
<tr>
<th>User</th>
<th>Authentication Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho$: Password (e.g., chars); $s$: Salt $H$: Secure Hash</td>
<td>${\text{UserID}, H(\rho, s), s}$</td>
</tr>
</tbody>
</table>

1. **Authentication Request**

2. **Image**

3. **UserID**

4. **Success or fail**

1. Recover $\rho'$ from $<x_1, y_1>, <x_2, y_2>, ...$
2. $H(\rho', s) = H(\rho, s)$?
   - Yes: success;
   - No: fail.

---

**Ganesh B. Gadekar, Prof. N. G. Pardeshi : Providing more security using graphical password- CaRP**
To recover a password successfully, each user-clicked point must belong to a single object or a clickable-point of an object. Objects in a CaRP image may overlap slightly with neighboring objects to resist segmentation. Users should not click inside an overlapping region to avoid ambiguity in identifying the clicked object. This is not a usability concern in practice since overlapping areas generally take a tiny portion of an object.

1.3 Typical application scenarios for CaRP include:

1. CaRP can be applied on touch-screen devices whereon typing passwords is cumbersome, esp. for secure Internet applications such as e-banks. Many e-banking systems have applied Captchas in user logins. For example, ICBC (www.icbc.com.cn), the largest bank in the world, requires solving a Captcha challenge for every online login attempt.

2. CaRP increases spammers operating cost and thus helps reduce spam emails. For an email service provider that deploys CaRP, a spam bot cannot log into an email account even if it knows the password. Instead, human involvement is compulsory to access an account.

![Working structure of CaRP](image-url)
2 Scope And Objective

I. **The proposed system will:**
   - Work as both a Captcha and a graphical password scheme and it provides reasonable security.
   - Be clicked-based graphical passwords which have large password space.
   - Does not reply on any specific Captcha scheme. When one Captcha scheme is broken, a new and more secure one may appear and be converted to a CaRP scheme.

II. **Objectives:**
   - Reasonable security and usability and appears to well with some practical applications for improving online security.
   - Protection against online guessing attacks, relay attacks, dictionary attacks etc.
   - To provide large password space than any existing security system.

3 Time Complexity

The total time complexity (T) can be calculated by summing the separate time complexities of all four processes i.e. q1, q2, q3,q4

\[ T = \sum_{i=1}^{4} T(q_i) \]

\[ T = T(q1) + T(q2) + T(q3) + T(q4) \]

\[ T = O(n^2) + O(n) + O(n) + O(1) \]

Therefore, the total time complexity is,

\[ T = O(n^2) \]

Here, Process q1 contained nested for loops, q2 & q3 contains a single for loop which executes for \( n \) times and q4 is simple comparison. Therefore, final time complexity is \( O(n^2) \):

Implementation Steps (Algorithm/Code)

1. Take user name.
2. Take the user's password.
3. Generate the CaRP image having user password and other non-password characters.

**Code**:

```java
public class DiscretizedCentralization {
    public static int[] applyDiscretizedCentralization(int x, int y, int radius) {
        int[] dcArray = new int[4];
        dcArray[0] = (x - radius) / (2 * radius);
        dcArray[1] = (y - radius) / (2 * radius);
        dcArray[2] = Math.abs((x - radius)) % (2 * radius);
        dcArray[3] = Math.abs((y - radius)) % (2 * radius);
        return dcArray;
    }

    public static int[] applyInverseDC(int x, int y, int radius, int xO_set, int yO_set) {
        int[] inverseDCArray = new int[2];
        inverseDCArray[0] = (x - xO_set) / (2 * radius);
        inverseDCArray[1] = (y - yO_set) / (2 * radius);
        return inverseDCArray;
    }
}
```
Ganesh B. Gadekar, Prof. N. G. Pardeshi: Providing more security using graphical password-CaRP

```java
return inverseDCArray;
public void getAlphabet(String alphabet) 
  jPasswordField1.setText(new String(jPasswordField1.getPassword()) + alphabet);
public char[] generateRandomAlphabets()
  Random r = new Random();
  char[] ch = new char[26];
  int temp;
  boolean ag;
  for (int i = 0; i < ch.length; i++) 
  while (true) 
  temp = r.nextInt(26) + 65;
  ag = true;
  for (int j = 0; j < i; j++) 
  if (ch[j] == temp) 
  ag = false;
  break;
  if (ag) 
  ch[i] = (char) temp;
  break;
  return ch;
public void GeneratePasswordCaptchaImage(char[] data)
  String[] type = f"Arial Black", "Serif", "Arial Black", "Vijaya",
  "SansSerif", "Kristen ITC", "Andalus", "SansSerif"g;
  int[] styles = fFont.BOLD + Font.ITALIC, Font.ITALIC, Font.BOLD,
  Font.ITALIC + Font.BOLD, Font.PLAIN + Font.ITALIC, Font.ITALIC,
  Font.BOLD, Font.ITALIC + Font.BOLDg;
  Color[] colours = fColor.BLACK, Color.BLUE, Color.YELLOW,
  Color.GREEN, Color.MAGENTA, Color.ORANGE, Color.PINK,
  Color.RED g;
  try 
  bu_eredImage = new Bu_eredImage(200, 200,
  Bu_eredImage.TYPEINTRGB);
  Graphics graphics = bu_eredImage.getGraphics();
  graphics.setColor(Color.WHITE);
  graphics._llRect(0, 0, 200, 200);
  graphics.setColor(Color.BLUE);
  Random rFont = new Random();
  Random rStyle = new Random();
  Random rType = new Random();
  Random rColors = new Random();
  int row = 1;
  int col = 1;
  char[] pass = selectedPassword.toCharArray();
  for (int i = 0; i <data.length; i++) 
```

Ganesh B. Gadekar, Prof. N. G. Pardeshi : Providing more security using graphical password- CaRP

5 Testing & Result (Analysis)

In the module to generate a CaRP image, User will give his/her username and will select password from the generated CaRP image if user already have signed up or he/she can sign up by giving username and password. Table 5.1 shows the analysis of designed module.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Task</th>
<th>Performed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>SignUp</td>
<td>Yes</td>
</tr>
<tr>
<td>2.</td>
<td>CaRP Image Generation</td>
<td>Yes</td>
</tr>
<tr>
<td>3.</td>
<td>Authentication</td>
<td>No</td>
</tr>
</tbody>
</table>

6 Conclusion

In this Article, I am implementing Captcha as Graphical Passwords which will be a new security primitive relying on unsolved hard AI problems. CaRP is both a Captcha and a graphical password scheme. The notion of CaRP introduces a new family of graphical passwords, which adopts a new approach to counter online guessing attacks: a new CaRP image, which is also a Captcha challenge. CaRP forces adversaries to resort to significantly less efficient and much more costly human-based attacks. In addition to offering protection from online guessing attacks, CaRP is also resistant to Captcha relay attacks, and, if combined with dual-view technologies, shoulder-surfing attacks. CaRP can also help reduce spam emails sent from a Web email service.

In the implemented module a CaRP image for particular user will get generated. User can sign up by giving his/her username and password. Password is displayed in CaRP image which is combination of password characters and non-password characters.

References


[10] Summers, Nick. "Vicarious claims its AI software can crack up to 90% of CAPTCHAAs offered by Google, Yahoo and PayPal". TNW


[18] Disabled Australian starts petition to kill CAPTCHA, an article by Tim Schiesser (August 5, 2013, 9:30 AM)


[21] Chellapilla, Kumar; Larson, Kevin; Simard, Patrice; Czerwinski, Mary, "Designing Human Friendly Human Interaction Proofs (HIPs)". Microsoft Research.


[23] The W3C paper Inaccessibility of CAPTCHA outlined some of the accessibility problems with CAPTCHAAs.

[24] The article Proposal for an accessible Captcha describes how audio and visual test can be combined to increase accessibility in a Captcha.