International Journal of Research
Available at https://edupediapublications.org/journals

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 09
August 2017

A New Security Primitive Based On Hard Ai Problems- *Captcha As A Graphical Password (Carp)*

E. Praganavi & A. Gayathri
Assistant Professor, Department of Computer Science, UCE OU, Hyderabad, Telangana, India.
pragnavi@gmail.com, & gayathri.aruta@gmail.com

Abstract— *CAPTCHA (Completely Automated Public Turing Test to tell computers and human apart) is a computer program which humans can pass but computer programs cannot pass. A new technology is built over the captcha called graphical captcha which eliminates dictionary attacks and hence more secure. With the hybrid use of CAPTCHA and graphical password, one can address a number of security problems such as relay attack and online guessing attacks. Shoulder surfing attacks can also be addressed with the help of dual view technology. CaRP (Captcha as a graphical password) is not a cure all to all attacks but it stipulates security and usability to legitimate and authenticated users in real time applications In the proposed concept, we present exemplary CaRPs built on both text Captcha and image-recognition Captcha. A user selects random and difficult images as passwords for reducing the guessing attack..*

Key Words: CAPTCHA, graphical password, CaRP, dictionary attacks, legitimate user.

I. INTRODUCTION

Authentication is a very necessary task in security where we use text password as a security technique. But text passwords are easy to break by many attacks [2]. Another traditional authentication approach for security is alphanumeric password which uses letters, upper and lower case characters and some special characters. Biometric is another way for security where human body part is used as a password with resilient to shoulder surfing attack [1, 2]. Nowadays, advanced technologies make use of Graphical Password, CAPTCHA, CaRP (captcha as a graphical password) for authentication and security purpose [3].

A. Graphical Password

Graphical password makes use of a picture, part of a picture or number of pictures together to authenticate legitimate user. These schemes are resistant to dictionary attacks but are prune to shoulder surfing attacks. Graphical password can be classified into three types based on memory capacity of the user: RECOGNITION, RECALL, and CUED RECALL. [2]

1) Recognition Based System: It is also known as search metric systems which generally requires user to memorize content here. Portfolio of images during password creation and then identify their images from among many random images to log in. Recognition-based systems use various types of images like faces, random art, everyday objects, and icons. Spoofing attacks are more difficult with recognition based systems. In recognition schemes, the system must know which images belong to a user's portfolio in order to display them. This current information must be stored such that its original form is available with the system.

2) Recall Based System: It is also known as draw metric systems because users recall and reproduce a secret drawing. In these systems, users typically draw their password either on a blank canvas or on a grid. Retrieval is done without memory prompts or cues. The system is prune to Phishing attacks. A phishing website attack can copy the login page from a authorized site, including the area for drawing the graphical password. Once user enters their username and password, this information can be used by attackers at the authorized site.

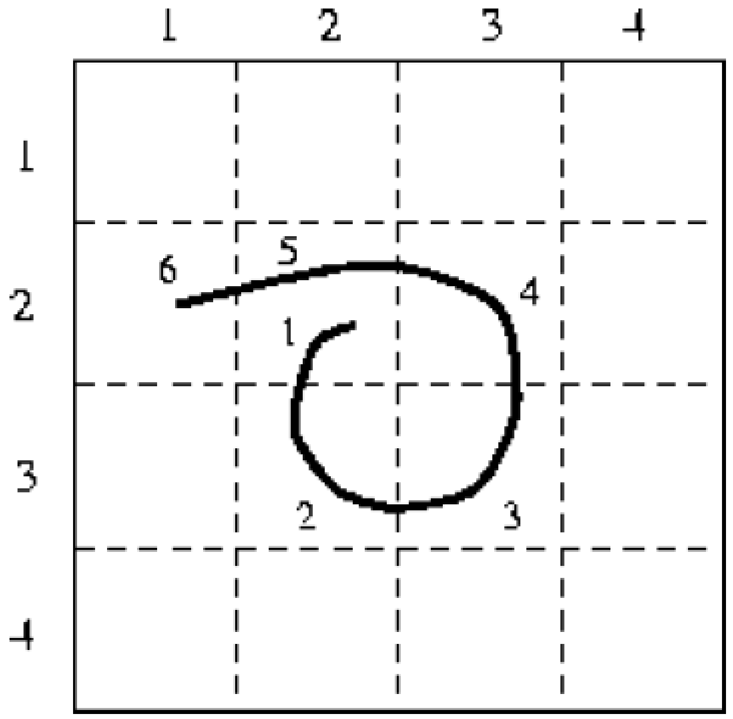


Figure 1: Draw A Secret

3) CUED Recall System: In cued recall scheme, users remember and then focus on specific location within the image. This system is also known as loci metric system which sometime also called as clicked based graphical password.

B. Captcha

Completely Automated Public Turing tests to tell Computers and Humans Apart abbreviated as CAPTCHA is a program that generates and grades tests that are human solvable, but beyond the capabilities of current computer programs. CAPTCHA is now almost a standard security mechanism for addressing undesirable or malicious Internet bot programs and major web sites such as Google, Yahoo and Microsoft all have their own CAPTCHAs. CAPTCHA is categorized into two: Text CAPTCHA which relies on character recognition and Image recognition CAPTCHA which deals with the recognition of non-character objects. It is used to prevent sensitive user inputs on an distrusted client which protects the communication channel between the user and web server from key loggers. But it fails to work good in front of online dictionary attacks [6].

II. CAPTCHA AS GRAPHICAL PASSWORD

CaRP ia a new way to thwart a guessing attacks. In a guessing attack, a password guess tested in failed trial is determined wrong and excluded from subsequent trials. The number of undetermined password guesses decreases with more trials, leading to a better chance of guessing the password [1]. Mathematically, let P be the set of password guesses before any trial, ρ be the password to find, A denote the attempts whereas An denote the n-th trial, and p(A = ρ) be the probability that ρ is tested in attempt A. Let Sn be the set of password guesses tested in trials up to An. The password guess to be tested in n-th attempt An is from set P|Sn−1, i.e., the relative complement of Sn−1 in P. If ρ ε P, then we have

$$p\,(A = \rho|A1 \;!= \rho, \ldots, An-1 \;!= \rho) > p(A = \rho)\ \text{(i)}$$

and

$$Sn \rightarrow P$$

$$p(A = \rho|A1 \;\_= \rho, \ldots, An-1 \;\_= \rho) \rightarrow 1 \text{ with } n \rightarrow |P|\ \text{(ii)}$$

CaRP falls for following two types of guessing attacks:

i. Automatic Guessing Attacks apply an automatic attempt and error process but P can be manually constructed.

ii. Human Guessing Attacks apply a manual attempt and error process. CaRP adopts a completely different approach to counter automatic guessing attacks. It aims at realizing the following equation in an automatic guessing attack.

CaRP falls for following two types of guessing attacks:

i. Automatic Guessing Attacks apply an automatic attempt and error process but P can be manually constructed.

ii. Human Guessing Attacks apply a manual attempt and error process. CaRP adopts a completely different approach to counter automatic guessing attacks. It aims at realizing the following equation in an automatic guessing attack.

$$p(A = \rho|A1, \ldots, An-1) = p(A = \rho), \forall n\ \text{(iii)}$$

Eq. (iii) means that each attempt is computationally independent of other attempt. Specifically, no matter how many attempts run previously, the chance of finding the password in the current attempt always remains the same. That is, a password in P can be found only probabilistically by automatic guessing (including brute-force) attacks, in contrast to existing graphical password schemes where a password can be found within a fixed number of trials.

A. Recognition based CaRP

In this system, infinite number of visual objects can be accessed as a password. Sequences of alphanumeric visual objects are also used in this system. ClickText, ClickAnimal, AnimalGrid are the 3 techniques used in CaRP [1].

Click Text: Clicktext is a novel technology for text CAPTCHA where characters can be arranged randomly on 2D space. It is different from text CAPTCHA challenge which is generally ordered from left to right sequence and user has to enter the data in that way. In ClickText, user click on the image which contains number of alphanumeric characters generated by CAPTCHA engine and user has to enter the password in same order.

Click Animal: This technology uses 3D models of animals to generate 2D animals with different textures, colors. These 2D animals as a result are then arranged on a background which is clustered. Some animals may be obstructed by other animals in the image but their essential parts are not obstructed so as to identify by the humans. It is a recognition based CaRP scheme developed on the top of Captcha Zoo.

Animal Grid: It is a combination of Click A Secret (CAS) and ClickAnimal. In this system, firstly ClickAnimal image is displayed, after the animal is selected, an image of n*n grid appears.

B. Recognition Recall CaRP

In this system, password is a sequence of some invariants points of objects. An invariant point of object is a point that has a fixed relative value in different fonts. User must identify the object image and then use identified objects as a cue to locate a password within a tolerance range. TextPoints and TextPoint4CR techniques are used in recognition recall CaRP [1].

III. APPROACH

The proposed system consists of mainly three different models that are user, server and trusted authority manager as shown in figure 2. User sends authentication request to the server with visual object IDs or clickable points of visual objects that user selects. Server request for the CaRP image through Trusted Authority Manager and records the location of the object from the image and sends that image to the user. Server calculates the coordinates sent by the user and authentication successes if the value matches.

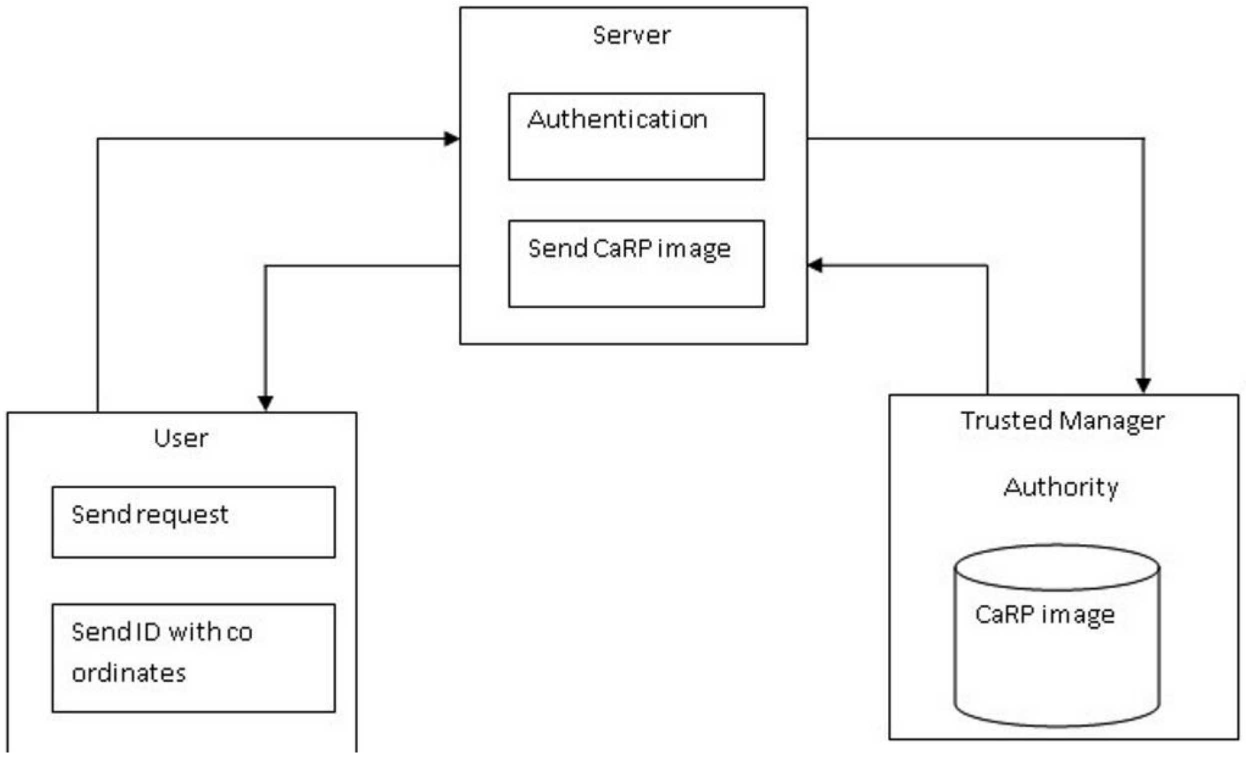


Figure 2: System Architecture

IV. IMPLEMENTATION

Nowadays Captcha is becoming more and more popular and receiving high demands. CaRP can offer users to use Software as a Service according to his need when the customer does not want to buy the software. System allocates different services depending on the type of user and provide the access rights. Front end of this system is JAVA, JSP and HTML. Back end that is Database is MySQL. Tomcat 7.0 is the Application Server used to host the cloud. Apache Tomcat is an open source servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the Java Server Pages (JSP) specifications from Oracle Corporation, and provides a "pure Java" HTTP web server environment for Java code to run. Initially we need to activate the Tomcat server. Like any social networking site you need to register yourself and create your account. Registration form will require your personal details and unique profile picture. Each User has to upload a unique profile picture. After selecting profile picture, user has to select captcha which is very essential for logging in. After registration the user can login to his account. After getting into your account the user will be provided with Upload Files, View files and change password options. Users are allowed to download and upload as many files as they desire. As we know all services cannot be provided free of cost. While uploading the file, user can upload it with or without security. If the user uploads with security, then the user has to select a particular point in the image which is used for verification. The x co-ordinate and y co-ordinate of the selected point is used to generate the hash value. In the view files option, we provide option to download file and delete file. To download the file which has security, user has to select the same point that was selected while uploading. Again hash value is generated for the selected point. If both the hash value matches, then the file can be further downloaded. Upon successful verification, user can download the file. If the file is uploaded without security, user can download it directly. Thus MD5 message-digest algorithm is used in our project for producing hash value and also to verify data integrity.

V. ALGORITHM

We are using MD5 algorithm with salt. A salt is random data that is used as an additional input to a one way function that hashes a password or passphrase.

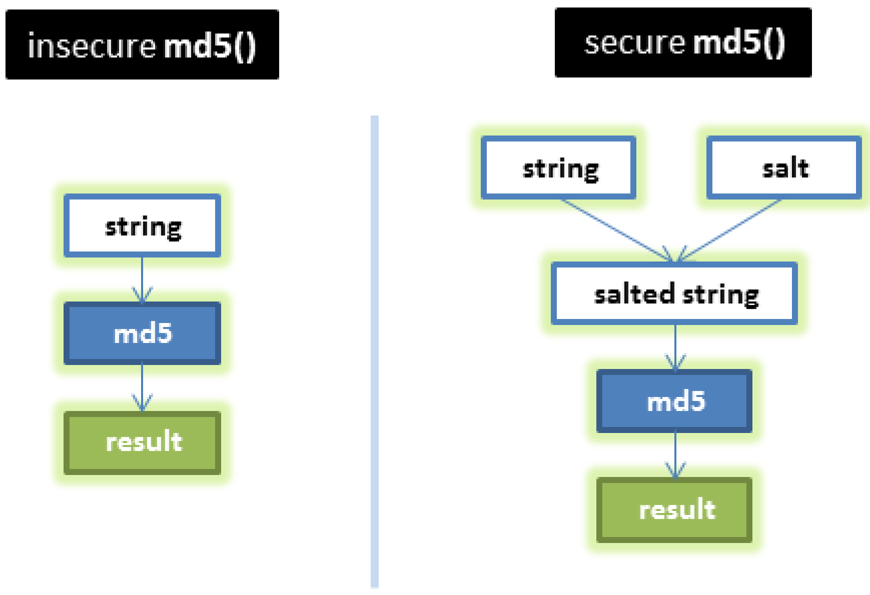
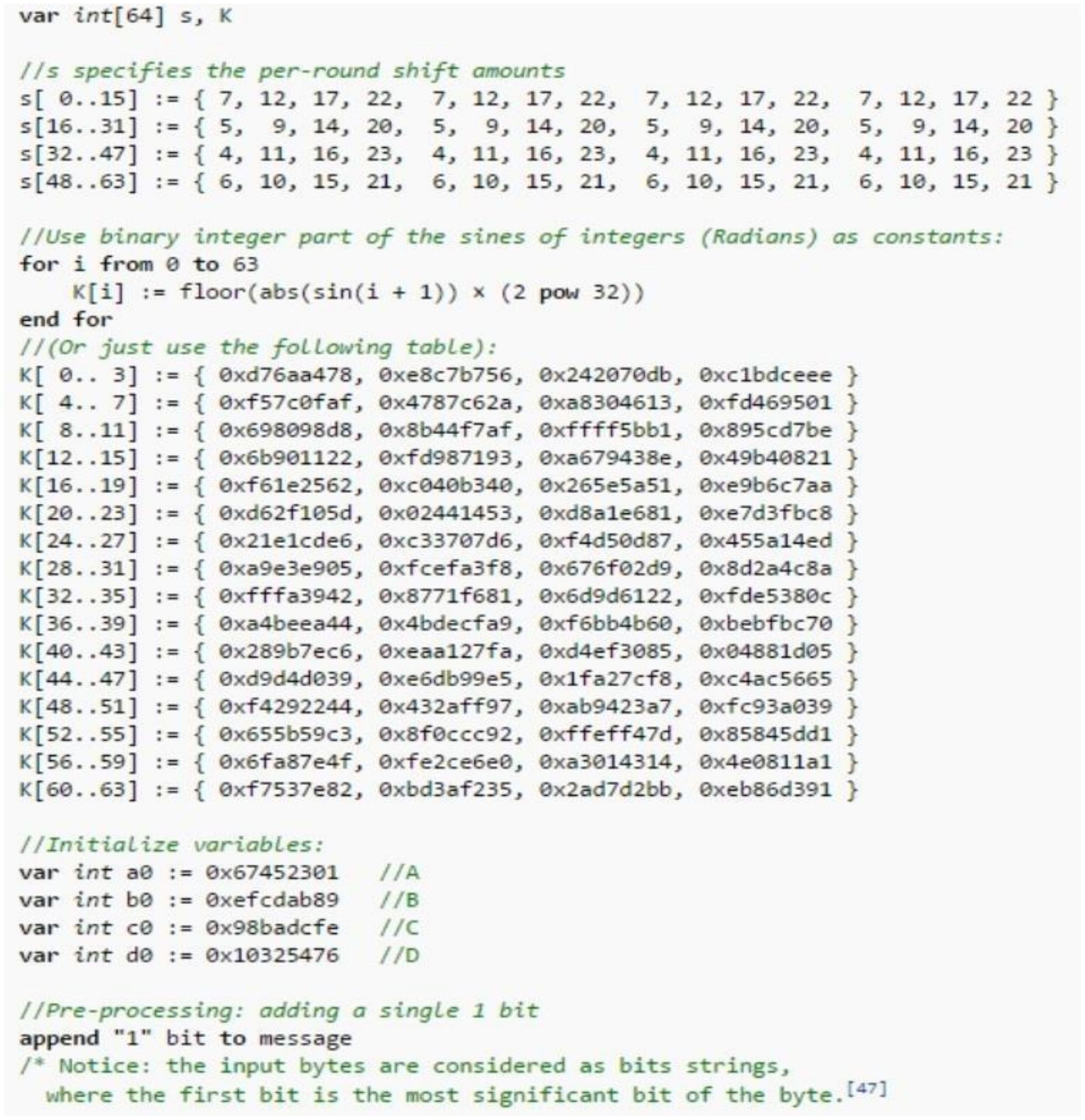


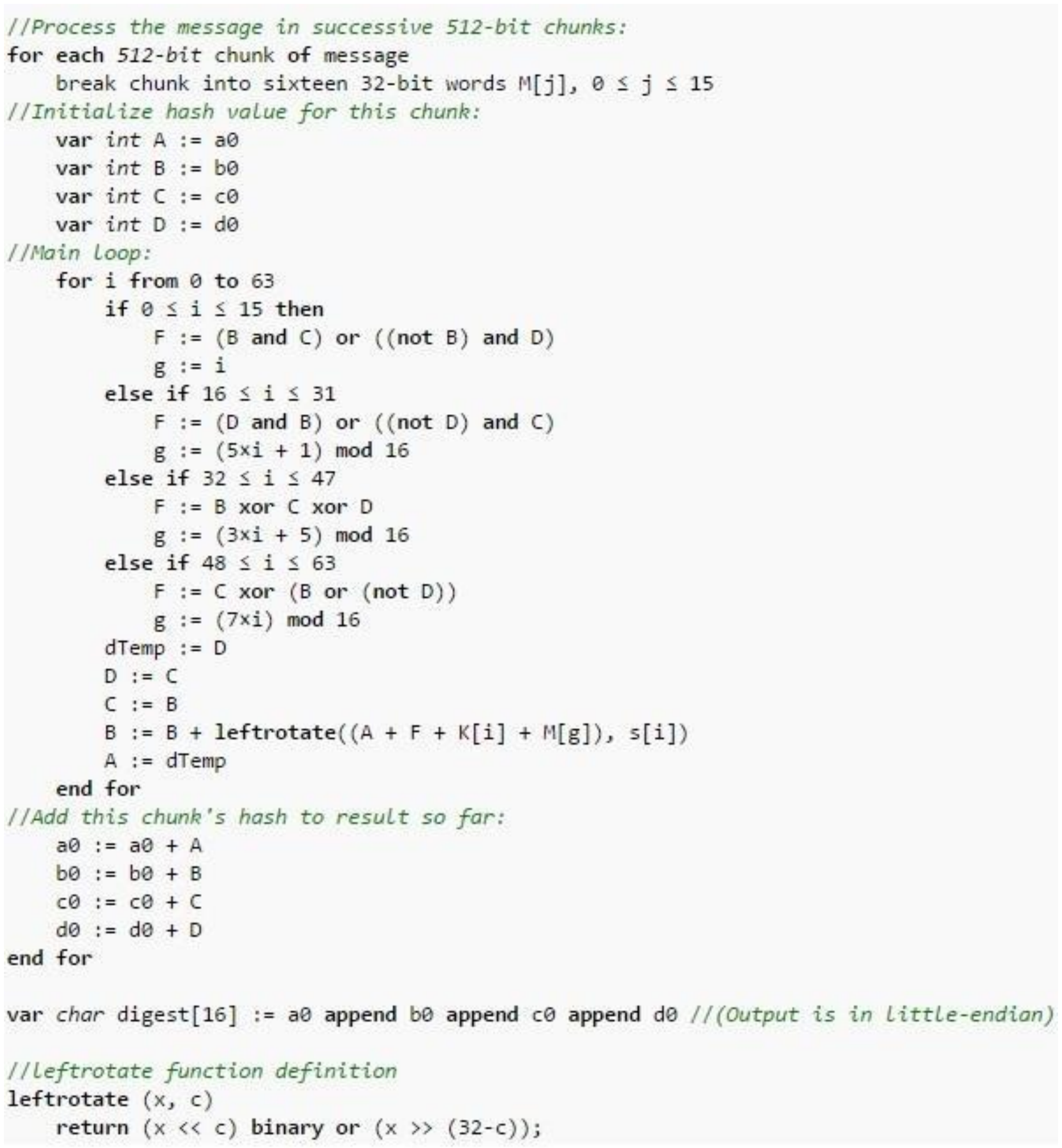
Figure 3: MD5 with Salt

MD5 processes a variable-length message into a fixed-length output of 128 bits. The input message is broken up into chunks of 512-bit block, the message is padded so that its length is divisible by 512. The padding works as follows: first a single bit, 1, is appended to the end of the message. This is followed by as many zeros as are required to bring the length of the message up to 64 bits fewer than a multiple of 512. The remaining bits are filled up with 64 bits representing the length of the original message, modulo 264. A salt is random data that is used as an additional input to a one way function that hashes a password. A new salt is randomly generated for each password. In a setting the salt and the password are concatenated and processed with a cryptographic hash function, and the resulting output is stored with the salt in a database. Hashing allows for authentication while defending against compromise of the plaintext password in the event that the database is somehow compromised.



```
var int[64] s, K

//s specifies the per-round shift amounts
s[ 0..15] := { 7, 12, 17, 22,  7, 12, 17, 22,  7, 12, 17, 22,  7, 12, 17, 22 }
s[16..31] := { 5,  9, 14, 20,  5,  9, 14, 20,  5,  9, 14, 20,  5,  9, 14, 20 }
s[32..47] := { 4, 11, 16, 23,  4, 11, 16, 23,  4, 11, 16, 23,  4, 11, 16, 23 }
s[48..63] := { 6, 10, 15, 21,  6, 10, 15, 21,  6, 10, 15, 21,  6, 10, 15, 21 }

//Use binary integer part of the sines of integers (Radians) as constants:
for i from 0 to 63
    K[i] := floor(abs(sin(i + 1)) × (2 pow 32))
end for
//(Or just use the following table):
K[ 0.. 3] := { 0xd76aa478, 0xe8c7b756, 0x242070db, 0xc1bdceee }
K[ 4.. 7] := { 0xf57c0faf, 0x4787c62a, 0xa8304613, 0xfd469501 }
K[ 8..11] := { 0x698098d8, 0x8b44f7af, 0xffff5bb1, 0x895cd7be }
K[12..15] := { 0x6b901122, 0xfd987193, 0xa679438e, 0x49b40821 }
K[16..19] := { 0xf61e2562, 0xc040b340, 0x265e5a51, 0xe9b6c7aa }
K[20..23] := { 0xd62f105d, 0x02441453, 0xd8a1e681, 0xe7d3fbc8 }
K[24..27] := { 0x21e1cde6, 0xc33707d6, 0xf4d50d87, 0x455a14ed }
K[28..31] := { 0xa9e3e905, 0xfcefa3f8, 0x676f02d9, 0x8d2a4c8a }
K[32..35] := { 0xfffa3942, 0x8771f681, 0x6d9d6122, 0xfde5380c }
K[36..39] := { 0xa4beea44, 0x4bdecfa9, 0xf6bb4b60, 0xbebfbc70 }
K[40..43] := { 0x289b7ec6, 0xeaa127fa, 0xd4ef3085, 0x04881d05 }
K[44..47] := { 0xd9d4d039, 0xe6db99e5, 0x1fa27cf8, 0xc4ac5665 }
K[48..51] := { 0xf4292244, 0x432aff97, 0xab9423a7, 0xfc93a039 }
K[52..55] := { 0x655b59c3, 0x8f0ccc92, 0xffeff47d, 0x85845dd1 }
K[56..59] := { 0x6fa87e4f, 0xfe2ce6e0, 0xa3014314, 0x4e0811a1 }
K[60..63] := { 0xf7537e82, 0xbd3af235, 0x2ad7d2bb, 0xeb86d391 }

//Initialize variables:
var int a0 := 0x67452301   //A
var int b0 := 0xefcdab89   //B
var int c0 := 0x98badcfe   //C
var int d0 := 0x10325476   //D

//Pre-processing: adding a single 1 bit
append "1" bit to message
/* Notice: the input bytes are considered as bits strings,
  where the first bit is the most significant bit of the byte.[47]
```

Figure 4a: MD5 Algorithm



```
//Process the message in successive 512-bit chunks:
for each 512-bit chunk of message
    break chunk into sixteen 32-bit words M[j], 0 ≤ j ≤ 15
//Initialize hash value for this chunk:
    var int A := a0
    var int B := b0
    var int C := c0
    var int D := d0
//Main loop:
    for i from 0 to 63
        if 0 ≤ i ≤ 15 then
            F := (B and C) or ((not B) and D)
            g := i
        else if 16 ≤ i ≤ 31
            F := (D and B) or ((not D) and C)
            g := (5×i + 1) mod 16
        else if 32 ≤ i ≤ 47
            F := B xor C xor D
            g := (3×i + 5) mod 16
        else if 48 ≤ i ≤ 63
            F := C xor (B or (not D))
            g := (7×i) mod 16
        dTemp := D
        D := C
        C := B
        B := B + leftrotate((A + F + K[i] + M[g]), s[i])
        A := dTemp
    end for
//Add this chunk's hash to result so far:
    a0 := a0 + A
    b0 := b0 + B
    c0 := c0 + C
    d0 := d0 + D
end for

var char digest[16] := a0 append b0 append c0 append d0 //(Output is in little-endian)

//leftrotate function definition
leftrotate (x, c)
    return (x << c) binary or (x >> (32-c));
```

Figure 4b: MD5 Algorithm

VI. CONCLUSION AND FUTURE SCOPE

We have proposed CaRP, 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, is used for every login attempt to make trials of an online guessing attack computationally independent of each other. A password of CaRP can be found only probabilistically by automatic online guessing attacks including brute-force attacks, a desired security property that other graphical password schemes lack. Hotspots in CaRP images can no longer be exploited to mount automatic online guessing attacks, an inherent vulnerability in many graphical password systems. 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. Our usability study of two CaRP schemes we have implemented is encouraging. For example, more participants considered AnimalGrid and ClickText easier to use than PassPoints and a combination of text password and Captcha. Both AnimalGrid and ClickText had better password memorability than the conventional text passwords. On the other hand, the usability of CaRP can be further improved by using images of different levels of difficulty based on the login history of the user and the machine used to log in. The optimal tradeoff between security and usability remains an open question for CaRP, and further studies are needed to refine CaRP for actual deployments. Like Captcha, CaRP utilizes unsolved AI problems. However, a password is much more valuable to attackers than a free email account that Captcha is typically used to protect. Therefore there are more incentives for attackers to hack CaRP than Captcha. That is, more efforts will be attracted to the following win-win game by CaRP than ordinary Captcha: If attackers succeed, they contribute to improving AI by providing solutions to open problems such as segmenting 2D texts. Otherwise, our system stays secure, contributing to practical security. As a framework, CaRP does not rely 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. Overall, our work is one step forward in the paradigm of using hard AI problems for security. Of reasonable security and usability and practical applications, CaRP has good potential for refinements. More importantly, we expect CaRP to inspire new inventions of such AI based security primitives.

REFERENCES

[1] Bin B. Zhu, Jeff Yan, Guanbo Bao, Maowei Yang, and Ning Xu "Captcha as Graphical Passwords—A New Security Primitive Based on Hard AI Problems" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 6, JUNE 2014 891

[2] R. Biddle, S. Chiasson, and P. C. van Oorschot, ―Graphical passwords: Learning from the first twelve years,‖ ACM Comput. Surveys, vol. 44,no. 4, 2012.

[3] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin, ―The design and analysis of graphical passwords,‖ in Proc. 8th USENIX SecuritySymp., 1999, pp. 1–15.

[4] S. Wiedenbeck, J. Waters, J. C. Birget, A. Brodskiy, and N. Memon, ―PassPoints: Design and longitudinal evaluation of a graphical password system,‖ Int. J. HCI, vol. 63, pp. 102 127, Jul. 2005.

[5] M. Alsaleh, M. Mannan, and P. C. van Oorschot, ―Revisiting defenses against large-scale online password guessing attacks,‖ IEEE Trans. Dependable Secure Comput., vol. 9, no. 1, pp. 128–141, Jan./Feb. 2012.

[6] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford, ―CAPTCHA: Using hard AI problems for security,‖ in Proc. Eurocrypt, 2003, pp. 294–311.

[7] B. Pinkas and T. Sander, ―Securing passwords against dictionary attacks,‖ in Proc. ACM CCS, 2002, pp. 161–170.

[8] P. Dunphy and J. Yan, ―Do background images improve ‗Draw a Secret' graphical passwords,‖ in Proc. ACM CCS, 2007, pp. 1–12.