iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 4 **Issue:** VII **Month of publication:** July 2016

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Graphical Authentication using Captcha (A New Security Initiative Based on Hard AI problem)

Waheeda Chandbadshah Akkalkot[1], Dr.Ramesh K[2], Dhananjay Potdar[3]
MCA, Dept of Computer Science, Karnataka State Women's University Vijayapur, Vijayapur, India
Dept.of Computer Science, Karnataka State Women's University, Vijayapur
M. Tech (CNE), Vijayapur, India

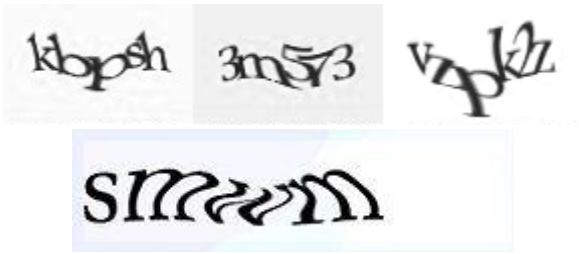
Abstract— Now a day's security is an very important issue. The traditional security systems are easily hackable. Using hard AI problems for security is emerging as an exciting new paradigm, but has been under-explored. In this paper, we present a new security method which works on hard AI problems; we provide a graphical password systems and Captcha technology together, which we call Graphical Authentication using Captcha (RAC). RAC is both a Captcha and a graphical password scheme. RAC addresses a number of security problems altogether, such as online guessing attacks, relay attacks, and. Notably, a RAC password can be found only probabilistically by automatic online guessing attacks even if the password is in the search set. RAC also offers a novel approach to address the well-known image hotspot problem in popular graphical password systems, such as PassPoints, that often leads to weak password choices. RAC is not a panacea, but it offers reasonable security and usability and appears to fit well with some practical applications for improving online security.

Index Terms—Component, formatting, style, styling, insert. (keywords)

I. INTRODUCTION

A. Captcha

A CAPTCHA is a type of challenge-response test used in computing to determine whether or not the user is human[1]. The most common type of CAPTCHA was first invented in 1997 by Mark D. Lillibridge, Martin Abadi, Krishna Bharat, and Andrei Z. Broder. This form of CAPTCHA requires that the user type the letters of a distorted image, sometimes with the addition of an obscured sequence of letters or digits that appears on the screen. Because the test is administered by a computer, in contrast to the standard Turing test that is administered by a human, a CAPTCHA is sometimes described as a reverse Turing test. This term is ambiguous because it could also mean a Turing test in which the participants are both attempting to prove they are the computer. Wherever we go on the internet, we encounter CAPTCHAs, those twisted words that block or enable entries on websites [2].



B. Graphical Passwords

A graphical password is an authentication system that works by having the user select from images, in a specific order, presented in a graphical user interface (GUI). For this reason, the graphical-password approach is sometimes called graphical user authentication (GUA). A graphical password is easier than a text-based password for most people to remember. Suppose an 8-character password is necessary to gain entry into a particular computer network. Instead of w8KiJ72c, a user might select any images.

Graphical passwords may offer better security than text-based passwords because many people, in an attempt to memorize text-based passwords, use plain words (rather than the recommended jumble of characters). A dictionary search can often hit on a password and allow a hacker to gain entry into a system in seconds. But if a series of selectable images is used on successive screen pages, and if there are many images on each page, a hacker must try every possible combination at random. If there are 100 images on each of the 8 pages in an 8-image password, there are 100^8, or 10 quadrillion (10,000,000,000,000,000), possible combinations that could form the graphical password! If the system has a built-in delay of only 0.1 second following the selection of each image

until the presentation of the next page, it would take (on average) millions of years to break into the system by hitting it with random image sequences.

The three types of password schemes being used to implement graphical elements into authentication are cognometric, locimetric, and drawmetric:

1) *Cognometric Scheme:* In cognometric schemes, the authenticator presents the user with a series of a set of images (often in a 3x3 or 4x4 grid), and the user is subsequently required to select the correct image from within each set. [6] Thus, this form relies on sequence. The number of sets presented is analogous to the length of a normal alpha-numeric password: the more sets a user is required to solve, the more complex the graphical password. Cognometric password is the initial stage when a user is either assigned or allowed to choose the objects that will form their password. During this initial stage, the user will be shown their objects slowly (and possibly one at a time) and be asked to remember something special about that object. Some have proposed using a Story method where the user creates a story that associates all the objects to aid memorability. An effective cognometric scheme will intentionally present a set of objects that have no extraordinary cues or distinguishing features that might prove useful as hints for potential hackers. [3]

2) *Locimetric Scheme:* In locimetric schemes, the user is presented with a single image, and he/she must click on regions of the image corresponding to his/her password. Because the recent Windows 8 release has been optimized for the touch screen, this scheme pairs nicely with the new operating system's gesture-oriented design. In Windows 8, users are given the option of signing in to their accounts using a picture password. To do this, users first choose their own picture and then create a password by using a combination of circles, straight lines, and taps on various locations of that picture. [3]

3) *Drawmetric Scheme:* Drawmetric schemes rely on users drawing shapes, figures, or doodles, but these are often difficult for computers to authenticate. In addition, this method relies on recall rather than pure recognition, the user having to completely recreate something (one of the weaknesses of traditional alpha-numeric passwords), and so this scheme is rarer than cognometric and locimetric schemes.

4) Hard AI problem: The most difficult problem are known as hard AI problem.
Hard AI problems are used for making computer as intelligent as people .
Hard AI problem can not be solved with modern computer alone, but would also require human computation. This property can be useful to test for the presence of humans.

II. BACKGROUND AND RELATED WORK

A. Graphical Passwords

A large number of graphical password schemes have been proposed. They can be classified into three categories according to the task involved in memorizing and entering passwords: recognition, recall, and cued recall. Each type will be briefly described here. More can be found in a recent review of graphical passwords [1].
A *recognition-based* scheme requires identifying among decoys the visual objects belonging to a password portfolio.
A typical scheme is Passfaces [2] wherein a user selects a portfolio of faces from a database in creating a password.
During authentication, a panel of candidate faces is presented for the user to select the face belonging to her portfolio. This process is repeated several rounds, each round with a different panel. A successful login requires correct selection in each round. The set of images in a panel remains the same between logins, but their locations are permuted. Story [20] is similar to Passfaces but the images in the portfolio are ordered, and a user must identify her portfolio images in the correct order.
A *recall-based* scheme requires a user to regenerate the same interaction result without cueing. Draw-A-Secret
(DAS) [3] was the first recall-based scheme proposed. A user draws her password on a 2D grid. The system encodes the sequence of grid cells along the drawing path as a user drawn password. Pass-Go [4] improves DAS's usability by encoding the grid intersection points rather than the grid cells.
BDAS [23] adds background images to DAS to encourage users to create more complex passwords.
In a *cued-recall* scheme, an external cue is provided to help memorize and enter a password. PassPoints [5] is a widely studied click-based cued-recall scheme wherein a user clicks a sequence of points anywhere on an image in creating a password, and re-clicks the same sequence during authentication.
Among the three types, recognition is considered the easiest for human memory whereas pure recall is the hardest [1].

B. Captcha

Captcha relies on the gap of capabilities between humans and bots in solving certain hard AI problems. There are two types of visual Captcha: text Captcha and Image-Recognition Captcha (IRC). The former relies on character recognition while the latter relies on recognition of non-character objects [26]–[30]. The following principle has been established: text
Captcha should rely on the difficulty of character segmentation, which is computationally expensive and combinatorially hard [30].
IRCs rely on the difficulty of object identification or classification, possibly combined with the difficulty of object segmentation.
Captcha can be circumvented through relay attacks whereby Captcha challenges are relayed to human solvers, whose answers are fed back to the targeted application.

C. Captcha in Authentication

It was introduced in [14] to use both Captcha and password in a user authentication protocol, which we call *Captcha-based Password Authentication (CbPA) protocol*, to counter online dictionary attacks. The CbPA-protocol in [14] requires solving a Captcha challenge after inputting a valid pair of user ID and password unless a valid browser cookie is received. For an invalid pair

of user ID and password, the user has a certain probability to solve a Captcha challenge before being denied access.
Applying a Captcha challenge only when the number of login attempts for the account has failed.
Captcha is an independent entity, used together with a text or graphical password. On the contrary, a RAC is both a Captcha and a graphical password scheme, which are intrinsically combined into a single entity.

III. RECOGNITION-BASED RAC

For this type of RAC, a password is a sequence of visual objects in the alphabet. Per view of traditional recognition based graphical passwords, recognition-based RAC seems to have access to an infinite number of different visual objects. We present two recognition-based RAC schemes and a variation next.

A. ClickText

ClickText is a recognition-based RAC scheme built on top of text Captcha. A ClickText password is a sequence of characters in the alphabet, e.g., ρ ="AB#9CD87", which is similar to a text password. A ClickText image is generated by the underlying Captcha engine as if a Captcha image were generated except that all the alphabet characters should appear in the image.
In ClickText images, characters can be arranged randomly

Fig. 2. A ClickText image with 33 characters.

Fig. 3. Captcha Zoo with horses circled red.

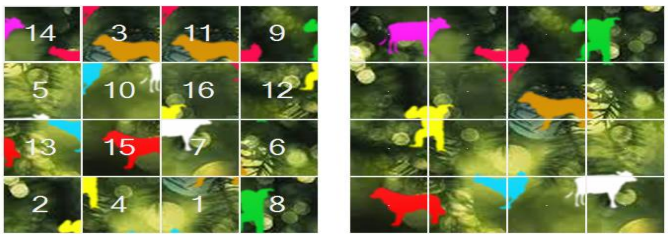


Fig. 4. A ClickAnimal image (left) and 6 × 6 grid (right) determined by red turkey's bounding rectangle on 2D space. This is different from text Captcha challenges in which characters are typically ordered from left to right in order for users to type them sequentially. Fig. 2 shows a ClickText image with an alphabet of 33 characters. In entering a password, the user clicks on this image the characters in her password, in the same order, for example "A", "B", "#", "9", "C", "D", "8", and then "7" for password ρ = "AB#9CD87".

B. ClickAnimal

Captcha Zoo [32] is a Captcha scheme which uses 3D models of horse and dog to generate 2D animals with different textures, colors, lightings and poses, and arranges them on a cluttered background. A user clicks all the horses in a challenge image to pass the test. Fig. 3 shows a sample challenge wherein all the horses are circled red.
ClickAnimal is a recognition-based RAC scheme built on top of Captcha Zoo [32], For each animal, one or more 3D models are built. The Captcha generation process is applied to generate ClickAnimal images: 3D models are used to generate 2D animals. The resulting 2D animals are then arranged on a cluttered background such as grassland. Some animals may be occluded by other animals in the image, but their core parts are not occluded in order for humans to identify each of them. Fig. 4 shows a ClickAnimal

image with many different shapes for the same animal's instantiations in the generated images. Combined with the additional anti-recognition mechanisms applied in the mapping step, these make it hard for computers to recognize animals in the generated image, yet humans can easily identify different instantiations of animals.

C. AnimalGrid

AnimalGrid's password space can be increased by combining it with a grid-based graphical password, with the grid depending on the size of the selected animal.

DAS [3] is a candidate but requires drawing on the grid.

To be consistent with ClickAnimal, we change from drawing to clicking: *Click-A-Secret (CAS)* wherein a user clicks the grid cells in her password. *AnimalGrid* is a combination of ClickAnimal and CAS. The number of grid-cells in a grid should be much larger than the alphabet size.

It has the advantage that a correct animal should be clicked in order for the clicked grid-cell(s) on the follow-up grid to be correct. If a wrong animal is clicked, the follow-up grid is wrong. A click on the correctly labeled grid-cell of the wrong grid would likely produce a wrong grid-cell at the authentication server side when the correct grid is used.

To enter a password, a ClickAnimal image is displayed first.

After an animal is selected, an image of $n \times n$ grid appears, with the grid-cell size equaling the bounding rectangle of the selected animal. Each grid-cell is labeled to help users identify.

Fig. 4 shows a 4 × 4 grid when the red turkey in the left image of Fig. 4 was selected. A user can select zero to multiple grid-cells matching her password. Therefore a password is a sequence of animals interleaving with grid-cells, e.g., ρ = "Dog, Grid_2_, Grid_1_; Cat, Horse, Grid_3_", where Grid_1_ means the grid-cell indexed as 1, and grid-cells after an animal means that the grid is determined by the bounding rectangle of the animal. A password must begin with an animal.

When a ClickAnimal image appears, the user clicks the animal on the image that matches the first animal in her password. The coordinates of the clicked point are recorded.

The bounding rectangle of the clicked animal is then found interactively as follows: a bounding rectangle is calculated and displayed, e.g., the white rectangle shown in Fig. 4. The user checks the displayed rectangle and corrects inaccurate edges by dragging if needed. This process is repeated until the user is satisfied with the accuracy of the bounding rectangle. In most cases, the calculated bounding rectangle is accurate enough without needing manual correction.

Once the bounding rectangle of the selected animal is identified, an image of $n \times n$ grid with the identified bounding rectangle as its grid-cell size is generated and displayed. The coordinates of user-clicked points on the grid image (the original one before scaling if the grid image is scaled) are recorded. The resulting sequence

of coordinates of user-clicked points, e.g., "AP_150,50_,

GP_30,66_, GP_89,160_, AP_135,97_,…" where "AP_x,y_" denotes the point with coordinates _x,y_ on a ClickAnimal image, and "GP_x,y_" denotes the point with coordinates _x,y_ on a grid image, is sent to the authentication server.

Using the ground truth, the server recovers the first animal from the received sequence, regenerates the grid image from the animal's bounding rectangle, and recovers the clicked grid-cells. This process is repeated to recover the password the user clicked. Its hash is then calculated and compared with the stored hash.

IV. SECURITY ANALYSIS

A. Automatic Online Guessing Attacks

In automatic online guessing attacks, the trial and error process is executed automatically whereas dictionaries can be constructed manually. If we ignore negligible probabilities, RAC with underlying CPA-secure Captcha has the following properties:

1) Internal object-points on one RAC image are *computationally-independent* of internal object-points on another RAC image. Particularly, clickable points on one image are computationally-independent of clickable points on another image. The first property can be proved by contradiction. Assume that the property does not hold, i.e., there exists an internal object-point α on one image A that is non-negligibly dependent of an internal object-point β on another image
2) An adversary can exploit this dependency to launch the following chosen-pixel attack. In the learning phase, image A is used to learn the object that contains point α. In the testing phase, point β on image B is used to query the oracle. Since point α is non-negligibly dependent of point β, this CPA-experiment would result in a success probability nonnegligibly higher than a random guess, which contradicts the CPA-secure assumption. We conclude that the first property holds.

The second property is a consequence of the first property since user-clicked internal object-points in one trial are computationally-independent of user-clicked internal object-points in another trial due to the first property. We have ignored background and boundary object-points since clicking any of them would lead to authentication failure.This is a great contrast to automatic online guessing attacks on existing graphical passwords which are deterministic, i.e., that each trial in a guessing attack can always determine if the tested password guess is the actual password or not, and all the password guesses can be determined by a limited number of trials. Particularly, brute-force attacks or dictionary attacks with the targeted password in the dictionary would always succeed in attacking existing graphical passwords.

B. Human Guessing Attacks

In human guessing attacks, humans are used to enter passwords in the trial and error process. Humans are much slower than computers in mounting guessing attacks. If we assume that 1000 people are employed to work 8 hours per day without any stop in a human guessing attack, and that each person takes 30 seconds to finish one trial

1) For 8-character passwords, the theoretical password space is $338 \approx 240$

2) for ClickText with an alphabet of 33 characters, $108 \approx 226$, , it take on average 2007 years_ It would take them on average $0.5 \cdot 338 \cdot 30/(3600 \cdot 8 \cdot 1000 \cdot 365) \approx 2007$ years to break a ClickText password

3) for ClickAnimal with an alphabet of 10 animals, and 3. $10 \times 467 \approx 242$, It would take $0.5 \cdot 108 \cdot 30/(3600 \cdot 8 \cdot 1000) \approx$ 52 days to break a ClickAnimal password

4) for AnimalGrid with the setting as ClickAnimal plus 6×6 grids. It would take $0.5 \cdot 10 \cdot 467 \cdot 30/(3600 \cdot 8 \cdot 1000 \cdot 365) \approx 6219$ years to break an AnimalGrid password[16].

C. Relay Attacks

Relay attacks may be executed in several ways. Captcha challenges can be relayed to a high-volume Website hacked or controlled by adversaries to have human surfers solve the challenges in order to continue surfing the Website, or relayed to sweatshops where humans are hired to solve Captcha challenges for small payments. Is RAC vulnerable to relay attacks? We make the same assumption as Van Oorschot and Stubblebine [8] in discussing CbPA-protocol's robustness to relay attacks: a person will not deliberately participate in relay attacks unless paid for the task. The task to perform and the image used in RAC are very different from those used to solve a Captcha challenge. This noticeable difference makes it hard for a person to mistakenly help test a password guess by attempting to solve a Captcha challenge. Therefore it would be unlikely to get a large number of unwitting people to mount human guessing attacks on RAC. In addition, human input obtained by performing a Captcha task on a RAC image is useless for testing a password guess. If sweatshops are hired to mount human guessing attack, we can make a rough estimation of the cost. We assume that the cost to click one password on a RAC image is the same as solving a Captcha challenge. Using the lowest retail price, $1, reported [14] to solve 1000 Captcha challenges, the average cost to break a 26-bit password is $0.5 \cdot 226 \cdot 1/1000$, or about 33.6 thousand US dollars.

D. Shoulder-Surfing Attacks

Shoulder-surfing attacks are a threat when graphical passwords are entered in a public place such as bank ATM machines. RAC is not robust to shoulder-surfing attacks by itself. However, combined with the following dual-view technology, RAC can thwart shoulder-surfing attacks. By exploiting the technical limitation that commonly-used LCDs show varying brightness and color depending on the viewing angle, the dual-view technology can use software alone to display two images on a LCD screen concurrently, one public image viewable at most view-angles, and the otherprivate image viewable only at a specific view-angle [15]. When a RAC image is displayed as the "private" image by the dual-view system, a shoulder-surfing attacker can capture userclicked points on the screen, but cannot capture the "private" RAC image that only the user can see. However, the obtained user-clicked points are useless for another login attempt, where a new, computationally-independent image will be used and thus the captured points will not represent the correct password on the new image anymore. To the contrary, common implementations of graphical password schemes such as PassPoints use a static input image in the same location of the screen for each login attempt.
Although this image can be hidden as the private image by the dual-view technology from being captured by a shouldersurfer, the user-clicked points captured in a successful login are still the valid password for next login attempt. That is, capturing the points alone is sufficient for an effective attack in this case. In general, the higher the correlation of user-clicked points between different

login attempts is, the less effective protection the dual-view technology would provide to thwart shouldersurfing attacks.

V. IMPLIMENTATIONS

ClickText and AnimalGrid were implemented using ASP.NET. ClickText was implemented by calling a configurable text Captcha engine commercially used by Microsoft. This Captcha engine accepts only capital letters. As a result, we chose the following 33 characters in our usability studies: capital letters except I, J, O, and Z, digits except 0 and 1, and three special characters "#", "@", and "&". The last three special characters were chosen to balance security and users' strong dislike of using non-alphanumeric characters in text passwords [16]. Characters were arranged in 5 rows. Each character was randomly rotated from −30◦ to 30◦ and scaled from 60% to 120%. Neighboring characters could overlap up to 3 pixels. Warping effect was set to the light level. Each image was set to 400 by 400 pixels. Fig. 2 in Section III-A shows an image generated with the above setting.

In our implementation of AnimalGrid, we used an alphabet of 10 animals: bird, cow, horse, dog, giraffe, pig, rabbit, camel, element, and dinosaur. Each animal had three 3D models. The number of animals in a ClickAnimal image ranged randomly from 10 to 12, with the extra animals randomly selected from the alphabet. In generating an animal object, one of the three 3D animal models was randomly selected, and posed at a random view in generating a 2D object. Each animal was assigned a color randomly selected from a set of 12 colors. Generated 2D objects were placed randomly on a grass background, with the main part of each animal not occluded by other animals. Each ClickAnimal image was also set to 400 by 400 pixels. A 4 × 4 grid was used for CAS. Cells were labeled clockwise starting from cell 0. Fig. 4 in Section III shows an example of generated ClickAnimal images and an example of grid images. There was a cross icon on top of a grid image that a user could click to close the grid image.

VI. CONCLUSION

We have proposed RAC, a new security primitive relying on unsolved hard AI problems. RAC is both a Captcha and a graphical password scheme. The notion of RAC introduces a new family of graphical passwords, which adopts a new approach to counter online guessing attacks: a new RAC 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. In addition to offering protection from online guessing attacks, RAC is also resistant to Captcha relay attacks, and, if combined with dual-view technologies, shoulder-surfing attacks. RAC can also help reduce spam emails sent from a Web email service.

Our usability study of two RAC 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 Like Captcha, RAC 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 RAC than Captcha. That is, more efforts will be attracted to the following win-win game by RAC 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, RAC 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 RAC 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, RAC has good potential for refinements, which call for useful future work. More importantly, we expect RAC to inspire new inventions of such AI based security primitives.

REFERENCES

[1] 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.
[2] (2012, Feb.). The Science Behind Passfaces [Online]. Available: http://www.realuser.com/published/ScienceBehindPassfaces.pdf
[3] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin, "The design and analysis of graphical passwords," in Proc. 8th USENIX Security Symp., 1999, pp. 1–15.
[4] H. Tao and C. Adams, "Pass-Go: A proposal to improve the usability of graphical passwords," Int. J. Netw. Security, vol. 7, no. 2, pp. 273–292, 2008. 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. ZHU et al.: NEW SECURITY PRIMITIVE BASED ON HARD AI PROBLEMS 903
[5] A. E. Dirik, N. Memon, and J.-C. Birget, "Modeling user choice in the passpoints graphical password scheme," in Proc. Symp. Usable Privacy Security, 2007, pp. 20–28. B. Pinkas and T. Sander, "Securing passwords against dictionary attacks," in Proc. ACM CCS, 2002, pp. 161–170.
[6] P. C. van Oorschot and S. Stubblebine, "On countering online dictionary attacks with login histories and humans-in-the-loop," ACM Trans. Inf. Syst. Security, vol. 9, no. 3, pp. 235–258, 2006. D. Davis, F. Monrose, and M. Reiter, "On user choice in graphical password schemes," in Proc. USENIX Security, 2004, pp.

1–11. P. Dunphy and J. Yan, “Do background images improve ‘Draw a Secret’ graphical passwords,” in Proc. ACM CCS, 2007, pp. 1–12. J. Yan and A. S. El Ahmad, “A low-cost attack on a Microsoft CAPTCHA,” in Proc. ACM CCS, 2008, pp. 543–554.

[7] G. Mori and J. Malik, “Recognizing objects in adversarial clutter,” in Proc. IEEE Comput. Society Conf. Comput. Vis. Pattern Recognit., Jun. 2003, pp. 134–141.

[8] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski, “Building segmentation based human-friendly human interaction proofs,” in Proc. 2nd Int. Workshop Human Interaction Proofs, 2005, pp. 1–10. R. Lin, S.-Y. Huang, G. B. Bell, and Y.-K. Lee, “A new CAPTCHA interface design for mobile devices,” in Proc. 12th Austral. User Inter. Conf., 2011, pp. 3–8.

[9] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, “Re: CAPTCHAs—Understanding CAPTCHA-Solving Services in an Economic Context,” in Proc. USENIX Security, 2010, pp. 435–452.

[10] S. Kim, X. Cao, H. Zhang, and D. Tan, “Enabling concurrent dual views on common LCD screens,” in Proc. ACM Annu. Conf. Human Factors Comput. Syst., 2012, pp. 2175–2184.

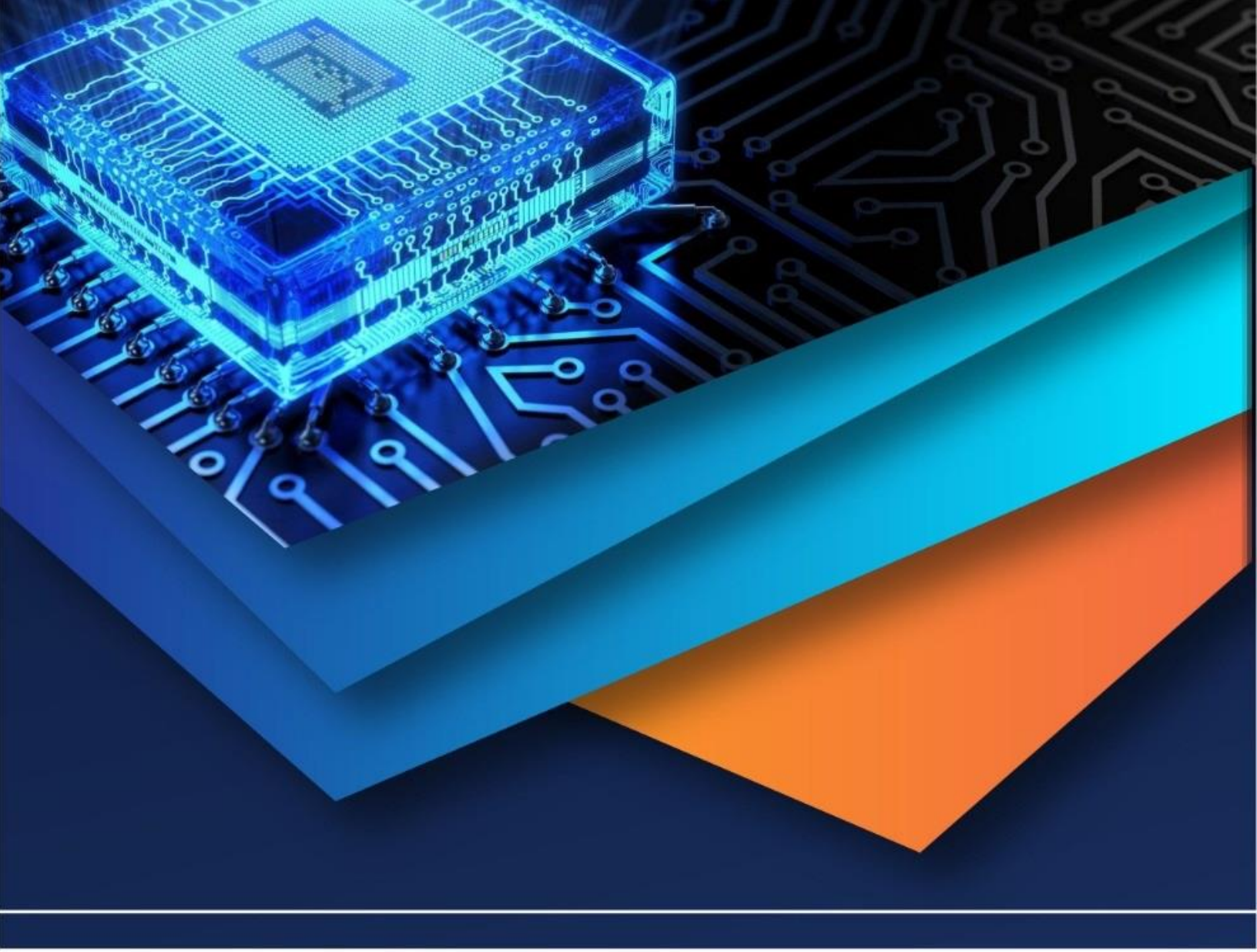
[11] J. Bonneau, “The science of guessing: Analyzing an anonymized corpus of 70 million passwords,” in Proc. IEEE Symp. Security Privacy, Jun. 2012, pp. 20–25.

AUTHORS DETAILS

Dr. Ramesh K **:** Associate Professor & Chairman in the Dept.of Computer Science at Karnataka State Women’s University,Vijayapur.Here I am working on various research problems related to the theory and practice of Multilayer Networks, Wireless Communication for Cochlear Implants.

Waheeda Chandbadshah Akkalkot
MCA
Dept of Computer Science, Karnataka State Women’s University Vijayapur, Vijayapur, India
Waheeda.akkalkot@gmail.com

Dhananjay Potdar
MTech(CNE), Vijayapur, India
Dhananjay.potdar@gmail.com



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY