RESEARCH ARTICLE OPEN ACCESS

Optimizing Security in Smartphones using Interactive CAPTCHA (iCAPTCHA)

Aisha Umar Suleiman, A. ArokiarajJovith
(Department of Information Technology, SRM University, Kattankulathur 603202, Chennai, India)
(Department of Information Technology, SRM University, Kattankulathur 603202, Chennai, India)

ABSTRACT
Websites such as email providers use Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) which is a simple test easily solved by humans but not by computers and hence provides a way of distinguishing a legitimate human user from an attacker. Methods have been developed like the Optical Character Recognition (OCR) and the third party human attack which have made CAPTCHAs vulnerable. The third party human attack poses a real threat to the use of CAPTCHA which can be easily shown using the Instant Messenger CAPTCHA Attack (IMCA).

A new defense system, the interactive CAPTCHA (iCAPTCHA) was developed to defend against third party human solver attacks. The iCAPTCHA is solved via user interactions and the back-and-forth traffic between the client and server amplifies the statistical timing difference between a third party human attacker and a legitimate human user.

The aim of this research is to use iCAPTCHA in Smartphones which will be solved after a certain number of password trials. However iCAPTCHA alone cannot provide sufficient security, therefore to further provide security, the iCAPTCHA will be encrypted using RSA and Elliptic Curve Cryptography.

***Keywords*–** attacks, CAPTCHA, cryptosystems, security, smartphones.

I. INTRODUCTION

Many companies have websites for online business and these companies offer a lot of free services to their users. Websites popular for the services they provide such as blogs, social networking, information searching and online gaming generally attract the attention of users. However, this advantage can be misused by attackers that use bots which are computer programs written that masquerade as normal users. They automatically register, access and exploit these services causing consumption of resources like memory and bandwidth which leads to downtime on the websites.

Websites such as email service providers use Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) which is a simple test easily solved by humans but not by computers and hence provides a way of distinguishing a human user from an attacker. It is based on hard Artificial Intelligence (AI) problems, which are speech recognition, natural language understanding and image recognition. It is mostly in the form of a distorted image containing short text displayed in such a format that only human eyes can recognize the alphabets clearly. Other forms of CAPTCHA are the image-based, audio-based and video-based CAPTCHAs. CAPTCHAs have now become a widely used standard security technology which helps prevent attacks on systems, networks or websites.[1]

However, methods have been developed which makes CAPTCHA vulnerable. The vulnerabilities can be classified into three categories which are the random guess attack, image processing techniques such as Optical Character Recognition (OCR) and third party human attack.[2]

It is needless to explain about the rise of smartphones in recent years. Anything once done on a desktop computer has now migrated to a smartphone in some form or another. A smartphone combines the functionality of a mobile device and a PDA and even though small in size and portable, it provides much computer functionality such as processing, communication and data storage with user input provision through a touchscreen or a small keyboard on the device.

Smartphones are increasingly becoming a target for security threats. According to FireEye Mobile Security Researchers, mobile versions of some websites (Amazon to be exact) provide an unlimited number of password entry trials when logging in to the site. This makes the site vulnerable to attacks like the brute-force attack by malicious users, whereby they try to access a legitimate user’s accounts by trying all possible entries until the right one is found.[3]

The Interactive CAPTCHA has been formulated as a defense against the categories of attack on the traditional CAPTCHA, especially the third party human attack and also against the brute-force attacks on password loggings for websites.[2]

II. INTERACTIVE CAPTCHA

Interactive CAPTCHA is an enhanced form of the traditional CAPTCHA technique created in order to provide more security that could not be met by normal CAPTCHA. As the name implies, iCAPTCHA involves a sequence of mouse clicks on a set of buttons provided to the user. A normal CAPTCHA image is first dynamically generated and displayed to the user to solve. As the user clicks on the image to begin the input sequence, a set of obfuscated characters appear on buttons which can be used to solve the CAPTCHA. One of the buttons will have the character corresponding to the first letter in the CAPTCHA image. Upon each click a new set of obfuscated buttons are displayed until the last character of the CAPTCHA image is solved.[3]

However in this study, we would show that the interactive CAPTCHA has a few drawbacks, for example the Man in the Middle Attack can be used to intercept the data as it is being sent from a user's browser to the server.

We would also propose a solution for this by using two encryption algorithms on the data as it is being sent from the user's browser to the server and determine which is more effective and efficient, that is a performance evaluation.

III. PROPOSED SYSTEM

The proposed system provides a security enhancement to the Interactive CAPTCHA by encrypting the data which is sent as the user response to the server.

The RSA and ECC encryption of the interactive CAPTCHA message are implemented on Android 4.4 as an operating system. The software platform being used is the Java EE 7 and Eclipse 4.4 (Luna) IDE.

To obtain the result, Secure Socket Layer (SSL) is used as the security protocol for the data that is encrypted using RSA or Elliptic Curve sent from the user to the server.

The selection of a cryptographic algorithm that would meet a user's needs should be based on attributes like security and performance. A cryptosystem design should take into account any positive and negative aspects of the algorithm in question and select the algorithmic tools that best address the problem to be solved.

The concept of public-key cryptosystems is based on the notion that keys come in pairs, one for encryption and a different but related one for encryption upon which the algorithms rely on. And it is computationally infeasible to decipher one key given the other key and the cryptographic algorithm.

Many algorithms have been proposed a lot of which have been proven insecure or impractical. Some may have a key too large or the cipher text is much larger than the plaintext. Of the secure and practical ones, some are only suitable for encryption only and others for key distribution. In this paper, we would be using two algorithms that work both ways, which are RSA and Elliptic Curve Cryptography.

2.1 RSA ALGORITHM

RSAis an algorithm for public-key encryption. It was the first algorithm known to be suitable for signing as well as encryption, and one of the first great advances in public key cryptography. RSA is widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations. The RSA scheme is a block cipher. Each plaintext block is an integer between 0 and $n - 1$ for some n, which leads to a block size $\leq \log2(n)$. The typical block size for RSA is 1024 bits. The RSA algorithm was invented by Ronald L. Rivest, Adi Shamir, and Leonard Adleman in 1977. The security of the algorithm is based on the hardness of factoring a large composite number and computing eth roots modulo a composite number for a specified odd integer. Its security comes from the computational difficulty of factoring large numbers. To be secure, very large numbers must be used for p and q - 100 decimal digits at the very least.[4]

Key Generation

- Select two prime numbers p and q.
- Calculate $n = pq$.
- Calculate $\varphi(n) = (p-1)(q-1)$.
- Select e such that it is relatively prime to $\varphi(n)$.
- Determine d such that $de = 1 \bmod \varphi(n)$.

Encryption

The message must be a number less than the smaller of p and q. However, at this point we don't know p or q, so in practice a lower bound on p and q must be published. This can be somewhat below their true value and so isn't a major security concern. It takes the following form:

$$C = M^e \bmod n \qquad \text{….(1)}$$

Decryption

This works very much like encryption, but involves a larger exponentiation, which is broken down into several steps. It is of the following form:

$$M = C^d \bmod n = (Me)d \bmod n = M^{ed} \bmod n \qquad \text{….(2)}$$

2.2 ELLIPTIC CURVE ALGORITHM

In 1985, Neal Koblitz and V. S. Miller independently proposed using elliptic curves for public-key cryptosystems. They did not invent a new cryptographic algorithm with elliptic curves over finite fields, but they implemented existing public-key algorithms, like Diffie-Hellman, using the elliptic curves.[4]

Key Generation

- At first we will take a curve in the form $Y^2=X^3+aX+b$, where a and$b$ are curve parameters.
- We then choose a prime number.
- Using point adding and point doubling we compute the points on the curve.
- Select a generating point out of those points whose order should be large.
- Then take a random number less than order of generating point as a private number for each entity. This will be a secret key.
- This entity will then generate its public key by multiplying the generating number with the secret number and will publish the point

Encryption

Each user A selects a private key nA and generates a public key PA= nA X G. To encrypt and send a message Pm to B, A chooses a random positive integer x and produces the cipher text Cm consisting of the pair of points. Encryption takes the following form:

$$Cm = \{xG, Pm + xPB\} \quad \text{….(3)}$$

(A uses B's public key PB to encrypt the message.)

Decryption

To decrypt the cipher text, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point:

$$Pm + xPB – nB(xG) = Pm + x(nBG) – nB(xG) = Pm \quad \text{….(4)}$$

IV. IMPLEMENTATION

To encrypt using RSA, the encryption/decryption key pair are generated from the values of p and q and user sends the values of ***N=pq*** and ***e*** (encryption key) to the server along with the HTTP request for the webpage. The server then stores the values so that when a message is sent, it can use the value of ***d*** which is calculated from ***N*** and ***e*** to decrypt the message

The RSA class has one constructor, two methods and two utility functions. The constructor creates the p and q values, calculates phi (φ) and the modulus. The next step is to create a public key from a random number between 0 and 2^{16}, and then generate the private key computing the modulus inverse function using the extended Euclid algorithm implemented within the BigInteger class. The code must be inside a loop because we want to make sure the public key generated has modInverse. The other functions are encrypt() which receives the message to encrypt and returns the BigInteger representing the encrypted text; and decrypt which receives the BigInteger and returns the decrypted message. The other two are utility functions to get the values of the private and the public key.

When using Elliptic Curve Cryptography, server picks a random integer ***a*** and computes ***aG*** which will be sent unencrypted but signed with its private key for authentication purpose, in a Server Key Exchange message.

The client checks that the signature is correct. It also picks a random integer ***b*** and sends ***bG*** in a Client Key Exchange. It will also compute ***b.aG = abG*** which is the premaster secret from which the master secret is derived.

ECC is a little difficult to implement. It creates a point and initializes it using the random() function, then creates the elliptic curve using the previously made point as G, then proceeds with creating the private and public keys. The encryption/decryption algorithm creates an EC point and packs it into a BigInteger secret, which would be the text to encrypt in a real world application the text would have been converted to EC points using already defined algorithms and then converted into BigIntegers each of those points. The cipher() function is called passing the public key and the secret to encrypt and it returns the pair of (kG,,Pm + kPb) which is also used by recover() in conjunction with the private key

As the Interactive CAPTCHA uses a timeout value for each button, which is calculated as the sum of the Round Trip Time (RTT), the time to encrypt the message E and the average time a user requires in solving the CAPTCHA U_{avg}, it is included in the client side script. This is done with the help of the ping command and pong response at the server and the client side respectively.

V. PERFORMANCE EVALUATION

RSA takes a message to the power of an exponent and the nature of that exponent determines the speed of the operation. In any public key system, the private key which is the exponent in RSA is used for signing and decrypting and is usually large so the calculation is quite slow. The public key is used for verifying and encrypting and the exponent can be quite small for RSA. To speed up verification and encryption, a small exponent can be used, however, this incurs security risks.[5]

With ECC, the time needed to generate a key pair is so short that even a device with the very limited computing power can generate the key pair the strength of the ECDLP algorithm means that strong security is achievable with proportionately smaller key and certificate sizes.

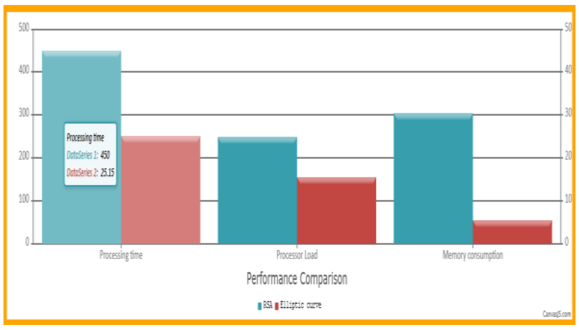

Figure 1 Processing Time

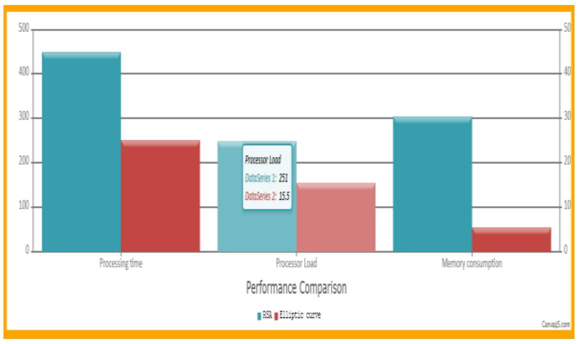

Figure 2 Processing Load

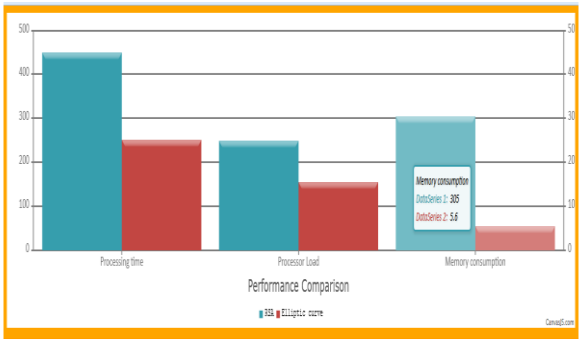

Figure 3 Memory Consumption

As shown depicted in the graphs in Fig. 1, Fig.2 and Fig. 3, the comparison of RSA and ECC in relation to performance, the processing time, processing load and memory consumption in ECC are less compared to RSA. We can say in this regard that it is due to the fact that ECC requires a lesser key size than RSA which makes generating that key a lot faster. And although the key is small, it does not make ECC any less secure than RSA, in fact, we can say that it has proven more secure.

VI. LIMITATIONS AND FUTURE WORK

The interactive CAPTCHA having provided solutions to some of the problems faced when using the traditional CAPTCHA unfortunately still has some limitations.

Users with impaired vision or motor skills*:* these legitimate users will likely have a slow response time U, and iCAPTCHA may reject their responses. Further evaluation is needed to accommodate impaired users.

Renegotiation attacks: The SSL connection is vulnerable to renegotiation attacks where an attacker can hijack an https connection to splice its own requests into the beginning of the conversation between the client and web server. The attacker can't actually decrypt the client-server communication, so it is different from a typical man-in-the-middle attack.

iCAPTCHA performance against character recognition based attacks: in order to break the current text-based CAPTCHA, an attacker algorithm needs to segment out each character of an image and identify it. iCAPTCHA may provide the attacker's algorithm some hints about each letter since it can match each character against a set of obfuscated candidate characters. The CAPTCHA image could be more distorted.

VII. CONCLUSION

Smart phones are becoming an integral part of today's society especially as technology has made possible the migration of anything once done on a computer to be done on a smart phone in one form or the other. We access a lot of websites through our smart phones as the mobile versions are now made available.

CAPTCHA is important for many websites in providing security, however, the strength of CAPTCHA alone has been made vulnerable with the third party human attack which uses the Instant Messenger technology.

A defense technology, the interactive CAPTCHA has been developed which involves multi-step back and forth traffic between client and server, which amplifies the statistical timing difference between a legitimate user and an attacker.

The concept of encrypting the interactive CAPTCHA response was introduced in this research work using two algorithms, RSA and ECC so as to prevent Man in the Middle attack. The two algorithms were compared and evaluated with respect to smart phones.

Since the crypto-sensitive operations of signing and decrypting can be much faster using ECC than RSA, ECC is more appropriate for use in applications that require much security. Also, websites using ECC need fewer server processing cycles allowing for more simultaneous SSL connections and faster page loading.

REFERENCES

[1] T.-E.Wei, A. Jeng, and H.-M. Lee, GeoCAPTCHA - A novel personalized CAPTCHA using Geographic Concept to Defend against 3rd Party Human Attack, *2012 IEEE 31st International Performance*

Computing and Communications Conference(IPCCC), 2012, pp. 392–399.

[2] H. Truong, C. Turner, and C. Zou, iCAPTCHA: The Next Generation of CAPTCHA Designed to Defend against 3rd Party Human Attacks,*2011 IEEE International Conference on Communications(ICC)*, 2011, pp. 1–6.

[3] M. Zheng, T. Wei and H. Xue, *MobileThreats, Threat Intelligence, Threat Reseach, Vulnerabilities*Amazon's Mobile Shopping Clients and CAPTCHA, 2014.

[4] Stallings, W. *Cryptography and Network Security* (Prentice Hall), 2003.

[5] S. Kancheti,*Comparative Study of Elliptic Curve Cryptography and RSA in Constrained Environment*, an unpublished master's thesis, Faculty of The Department of Computing Sciences Texas A&M University-Corpus Christi Corpus Christi, Texas.