Scientific Research Publishing

World Journal of Engineering and Technology, 2024, 12, 274-285
https://www.scirp.org/journal/wjet
ISSN Online: 2331-4249
ISSN Print: 2331-4222

Sensitive Information Security Based on Elliptic Curves

Nadine Nibigira[1], Vincent Havyarimana[2], Zhu Xiao[3]

[1]School of Engineering Sciences, University of Burundi Doctoral School, Bujumbura, Burundi
[2]Department of Applied Sciences, Burundi Higher Institute of Education (ENS), Bujumbura, Burundi
[3]College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
Email: nibinadine@gmail.com, havincent12@gmail.com, zhu.xiao.work@gmail.com

How to cite this paper: Nibigira, N., Havyarimana, V. and Xiao, Z. (2024) Sensitive Information Security Based on Elliptic Curves. *World Journal of Engineering and Technology*, **12**, 274-285.
https://doi.org/10.4236/wjet.2024.122018

Received: February 28, 2024
Accepted: April 20, 2024
Published: April 23, 2024

Abstract

The elliptic curve cryptography algorithm represents a major advancement in the field of computer security. This innovative algorithm uses elliptic curves to encrypt and secure data, providing an exceptional level of security while optimizing the efficiency of computer resources. This study focuses on how elliptic curves cryptography helps to protect sensitive data. Text is encrypted using the elliptic curve technique because it provides great security with a smaller key on devices with limited resources, such as mobile phones. The elliptic curves cryptography of this study is better than using a 256-bit RSA key. To achieve equivalent protection by using the elliptic curves cryptography, several Python libraries such as cryptography, pycryptodome, pyQt5, secp256k1, etc. were used. These technologies are used to develop a software based on elliptic curves. If built, the software helps to encrypt and decrypt data such as a text messages and it offers the authentication for the communication.

Keywords

Cryptography, Elliptic Curves, Digital Security, Data, Sensitive Data, Implementation

1. Introduction

During the 20th century, a field of mathematics called number theory revealed a valuable gem for modern cryptography [1] elliptic curves. These seemingly simple curves are defined by elegant mathematical equations.

The cryptography is a fundamental area of computer security that aims to protect data and ensure the confidentiality, integrity, and authenticity of information exchanged between parties [1] [2]. Over the decades, cryptography has

evolved to adapt to the needs of modern digital society, and elliptic curves have become one of the pillars of this evolution.

Several researchers have found vulnerabilities in RSA public keys in recent years. In 2020, Keyfactor researchers analyzed more than 75 million active RSA keys across the internet [3], discovering that 1 in every 172 certificates using RSA keys is vulnerable as its used by other certificates to a practical attack known as "factoring" [4]. The foundation of a factoring attack is the ability to compute the private key d given the discovery of p and q by factoring n.

Therefore, calculations like RSA are customary calculations with tremendous key sizes, which require higher memory points of confinement and high get-ready forces. In light of these deductions, the use of RSA for giving any text, like SMS security, will diminish the execution of the contraption, and moreover, the taking care of time will be high [5] [6].

While ECC offers significant advantages, it is essential to note that the security of any cryptographic system depends not only on the algorithm itself but also on the implementation, key management, and other security practices. It is crucial to follow recommended guidelines and best practices to ensure the overall security of sensitive information.

The elliptic curves are particularly interesting for their combination of geometrical elegance and structures rich in arithmetic [7]. These properties make the elliptic curves a powerful tool in various fields [8], from cryptography to the theory of the numbers passing by to algebraic geometry.

The most important reason for using elliptic curves for secure SMS is that both the sender and the receiver are using comparable technology and should change the key. This ensures complete verification and ensures that the SMS can be decrypted at the exact time of transmission and is therefore protected from subsequent decryption attempts [9] by other people. In this way, ECC can be efficiently used as part of securing and confirming the correspondence between the two parties [10]. The bitsize of ECC seems to be different than that of RSA and moreover, due to the discrete logarithmic problem, it is very difficult to decrypt the messages knowing only the open key [15]. ECC consumes less key size, and now scheduling performance in this way can be extremely beneficial in small devices such as cell phones [11]. The study chose ECC, which is better than using a 256-bit RSA key. In order to achieve equivalent protection through the use of ECC, several Python libraries were used. The cryptography software was developed and is presented in this article. This research study details that by using Python, ECC provides a level of security that requires fewer computational resources to encrypt and decrypt data compared to alternative methods such as RSA. In addition, communication transmission protection can be usefully used as small devices such as telephones [12].

The next sections of this work are structured as follows: The second section shows the applications of elliptic curves in cryptography. One of the main concerns of cryptography is to ensure that an attacker cannot understand or alter the data even if they intercept it. Sensitive information security is all about pro-

tecting data that could be misused if it falls into the wrong hands. This data is valuable because it is private, confidential, or could be used to cause harm. The third part is about the methodology used in this study and the Python libraries and their main tasks are described in detail. In the last section, the software developed in this article is introduced using some interfaces.

2. Applications of the Elliptic Curves in Cryptography

The elliptic curves play a crucial role in the security of many applications in cryptography. There are many, such as the security of financial transactions and government communications [10] [13].

The following shows the elliptic presentation (**Figure 1**) and its equation.

$$y^2 = x^3 + ax + b \tag{1}$$

Elliptic curves are powerful mathematical tools that play a central role in securing communications and protecting privacy and data confidentiality [12]. Their ability to provide robust security with relatively small key sizes makes them indispensable in many cryptographic applications, from financial transaction security to biometric authentication of defense department personnel. Their use, combined with a careful selection of parameters, guarantees a level of security in an increasingly networked digital world. Three main algorithms are used for PKI key generation: Rivest Shamir Adleman (RSA), Digital Signature Algorithm (DSA), and Elliptic Curve Cryptography (ECC) [14].

2.1. Security of the Transactions Financial

An arrangement between two parties wherein one of them provides property as security or collateral for a loan is known as a secured transaction [15]. The elliptic curves are widely used to ensure the confidentiality and integrity of financial transactions.

2.2. Diplomacy and Security National

Protocols based on ECDH are used by governments to establish secure communications between diplomats, national security agencies, and the military [16]. These protocols allow the parties to ensure that their communications remain

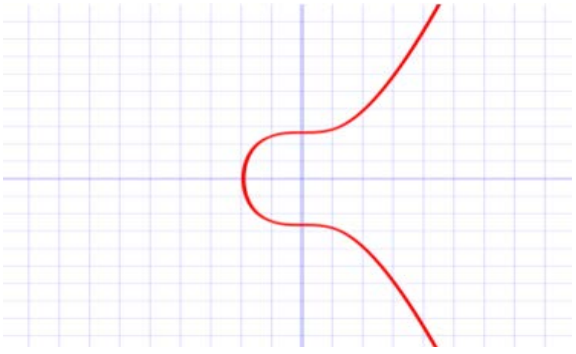

Figure 1. Elliptic curves presentation.

confidential and are not compromised by malicious actors.

2.3. Authentication Biometric

The elliptic curves are also used in authentication biometrics systems [17], which allow verifying a person's identity based on their physiological or behavioral characteristics. The fingerprint is often stored underneath the form of cryptography. The elliptic curves can be used to biometrically secure this data [18] and prevent its access.

2.4. Security of the Objects Connected (IoT)

The connected objects such as medical devices and thermostats require security mechanisms to protect sensitive data and ensure the integrity of their operation [19]. The elliptic curves are suitable for this use due to their low power consumption for computation and storage.

3. Methodology Used

3.1. Choice of Python Libraries

In a practical context, the study conducted an in-depth comparative study of the available Python libraries for implementing elliptic curve-based cryptography. Our goal was to evaluate different libraries based on various criteria such as usability, documentation, performance and flexibility. After an in-depth analysis of various existing libraries, the study identified the following Python libraries as the most relevant to this study.

3.1.1. Cryptography

Cryptography is a comprehensive library that supports elliptic curve-based cryptography, providing a wide range of functionality. It provides various high-level interfaces for key generation, key exchange, digital signatures, encryption and decryption. The library follows best practices and security standards, making it a reliable choice for implementing elliptic curve-based cryptography. It is regularly maintained and updated to address security vulnerabilities and provide news.

3.1.2. Pycryptodome

Pycryptodome is a powerful Python library that supports elliptic curve-based cryptography as well as other cryptographic algorithms. It provides basic cryptographic primitives and algorithms that enable precise control of curve operations and elliptical trainers. It offers a wide range of elliptic curves, including popular curves such as secp256k1, providing flexibility in choosing the curve you want. The library focuses on performance, with implementations optimized for operations effective on the elliptic curves.

3.1.3. PyQt5

Although PyQt5 is best known for its capabilities in developing interface graphics, it can also be used to create user-friendly interfaces for cryptographic applications. It provides a robust framework for the design and interface graphics,

improving the usability and accessibility of these cryptographic solutions. PyQt5 creates interactive interfaces that allow users to easily perform various cryptographic operations based on the elliptic curves.

3.1.4. Secp256k1

Secp256k1 is a library specifically designed for elliptic curves. Secp256k1 is commonly used in cryptographic apps such as Bitcoin. It proposes implementations optimized for the operations of the elliptic curves, which makes it very efficient for tasks cryptographically related to secp256k1. The secp256k1 only focuses on the secp256k1 curve, which may limit its applicability if it needs to adopt other curves.

3.1.5. Libnum

Libnum is a Python library that provides functionality and mathematics for elliptic curve-based cryptography. It provides the tools to perform arithmetic operations on large numbers as a whole. This is essential for elliptic curve calculations. The libnum library provides functions such as prime number generation, Legendre symbol calculation, modular square root calculation, etc. Although libnum is not specifically aimed at elliptic curves, it provides useful mathematical functions that can be used in conjunction with other libraries to implement operations on elliptic curves.

In the frame of this research, it has taken in account several criteria of selection, such as:

- **Ease of use:** The libraries offering of the user-friendly interfaces and a reduced learning curve for our team of development;
- **Documentation:** The libraries offering with a comprehensive and well-structured documentation, providing examples and of the guides detailed for facilitate the integration and use;
- **Performance:** It granted a attention particular to performance of the libraries, ensuring that they meet our requirements of speed and efficiency for the operations cryptographic;
- **Flexibility:** The capacity of libraries to take in charge different elliptic curves and algorithms cryptographic, in order to answer has our needs specific.

3.2. Data Process

Preparation of Message

Before proceeding with encryption, it is important to prepare a clear message. To complete the work successfully, this work contains steps to prepare the message.

3.3. Subdivision in Blocks

In some cases, the message may be too long to be directly encrypted once. Therefore, it is common practice to divide it into fixed-size blocks before encryption. Each block can then be processed individually during the encryption process. Here the clear message is divided into blocks of two characters each, *i.e.*

a combination of bigrams.

3.4. Diluting (Padding)

Once the size of the final block is smaller than the set fixed size, it needs to be filled with additional data to reach the required size. Various filling methods can be used, e.g. B. padding with zeros or padding with special bytes to indicate the end of a message.

3.5. Scanning of Message

Before encryption, the simple message must be converted into a binary representation. This can be achieved through the bias of a standardized encoding such as UTF-8, which allows a wide range of characters to be represented. UTF-8 assigns a numeric value to each character in the sentence, making it easier to handle during encryption. In our case, we decided to convert the clear message into a sequence of numeric values taken from the ASCII code for each letter of the message.

3.6. Cutting in Blocks

In cutting the message in of the blocks of size fixed of 2 characters, the word "Hello" will be so divided in 3 blocks: "Hey", "ll" and "o".

If the size of the last block is less than 2 characters, we need to fill it for reach the size required. For the word "Hello" the size of third block is less than the required size, so we can pad it with zeros for reach 2 characters: "o0".

3.7. Scanning

All the blocks is converted to a torque sequence using the Standard ASCII. Each character is represented by a numerical value corresponding. Here is the result of this stage:

Hey => (72, 101);

ll => (108, 108);

o0 => (111.48).

After carrying out this step of preparation, the message is ready as show on **Figure 2** below. The output of this **Figure 2** is computed by using the algorithm based on elliptic curves with the keys appropriate.

It is important to note the specific details of the preparation of a message in terms of the encryption algorithm used, the constraints of security, and the requirements specific to the application. However, the fundamental principles of encoding, cutting in blocks, and filling remain generally the same.

3.8. Encryption

Once public keys have been exchanged, the sender can encrypt a message with the recipient's public key [14]. The encryption process generally involves a scalar multiplication between the recipient's public key and a point representing the

(142939024599300379614973397994676948258975798495
857435385489623711473352892 00,6809913612188444357
246137503528718446910182634153950439843288310949
5457523040)(214408536898950569422460096992015422 3
884636977437861530782344355672100293380 0,74708674
716484828384384391912455957923426546430173695522
937533403436221406 29)(606337960381642069984987271
937419988076820971566582093017488449914884201280
71,536767411341104521403836790234896170584437427 0
858613795895707536361685159944 4)

Figure 2. The cryptogram of the word "Hello".

message on the elliptic curves. This operation generates a dot cipher that represents the message in Figure 2. After the message preparation is complete, the message is now represented in the form of a series of dots. At this point the message can be encrypted. The principle of encrypting a message using elliptic curve-based cryptography is to convert the message into the form of a sequence of points on an elliptic curve.

Continuing with the previous example, we can try to explain the process encryption:

Message clear: "Hello" will be crypted by the elliptic curves and the keys private:

k1 = 998576006824493514398863100559297413588111297399950755193671 3538561210573425 7;

k2 = 833120098185547895337288018839843487958872113879993546440855 87393550585247937.

3.9. Calculation of the Key Public with K1

To find the public key, we calculate the scalar multiplication of G with k1 or G with K2 modulo (4.2). Using G and K1 we find the point P with as the contact details:

P (618207055952428391705191846689360367960850012413630153409576174 03010458729107, 167249372734477413890767353760258720661779400716094692 8412015038510113137691).

3.10. Calculation of the Key Secret Common

In previous section 3.9 subsection, the point P has been calculated and is ready. Each part of results calculated its present a common secret key and that key it is used as the public key public. To the case of this study, k1 was used for generate P, and P is used as well K2. The calculation through the secret key always compute the multiplication scalar of P and K2. The contact details of point P become:

P (26666503711074136871009844561213910012568683979549540475470976497522348004 42, 87525029468054329130497636196949977048432243866840364524585935905663425575722).

The point found at this step is the point used by the sender when encryption of a message. It can now perform encryption of the word "Hello" using the settings calculated here.

3.11. Decryption

The recipient can decrypt the encrypted message using their own key secret common.

Calculation of the key of decryption

To decrypt, the recipient must calculate the modular inverse of their key common secret multiply by the modulo, here (4.2). $P * (4.2) \equiv 1$ modulo n. Here we look for not.

The decryption process involves a scalar multiplication between the key private of recipient and the point encrypting. This operation reverses the encryption and allow recover the clear message.

3.12. The Electronic Signature

Cryptography based on ECC elliptic curves is used to securely create and verify digital signatures. The following are the steps to generate a signature using the private key and verify it using of the public key in a signature scheme ECC:

1) Generation of the pair of keys

- ✓ The signer generates a key pair, including a private key and a public key, using a specified elliptic curves (for example, secp256k1).
- ✓ The private key is a random number chosen from a certain interval defined by the elliptic curves.
- ✓ The public key is obtained by performing scalar multiplication between the key private and a point of base fixed on the elliptic curves.

2) Generation of the signature

The signature to be used in a given message:

- ✓ The message is generally transformed in a value of hash has ugly of a function of hash cryptographic, such that SHA-256.
- ✓ The signature used her key private for generate a signature digital of the message.
- ✓ For generate the signature, the signatory follows an algorithm of signature specific, such that the algorithm ECDSA.
- ✓ The algorithm ECDSA used of the operations mathematics based on elliptic curves to produce the digital signature.

3) Verification of the signature

- ✓ The recipient or a third party wishing check authenticity of the signature used the key public corresponding to the signature.
- ✓ The received message is also transformed into a hash value to ugly of the

even function of hash cryptographic that that used by the signature.

- ✓ The signature verification algorithm (e.g. ECDSA) is applied using the key public, the signature and the message chopped.
- ✓ The verification algorithm performs mathematical operations on the elliptic curves for to validate the digital signature.
- ✓ If the verification is successful, this confirms the authenticity of the signature and integrity of message.

4. Presenting the ECC Software

This software is built with python 3.11 and the editor is Visual Studio Code.

4.1. The Main Interface

The main interface of the application of elliptic curve-based cryptography forms the central point of the user's interaction with the system. It offers easy access and has all the features and operations available. This interface allows the user to perform various tasks related to elliptic curve-based cryptography in an intuitive manner.

1) Menu of navigation

The navigation menu is located on the left side of the interface and provides options for accessing the various app features, such as: E.g., generating the points of an elliptic curve, generating the numbers first, scalar multiplication, doubling a point, calculating two points, etc. The user can select the desired operation by clicking on the corresponding option in the menu.

2) Area display of the results

This area displays the results of the operations performed by the user. It can display information such as the points generated, the first numbers calculated, the results of the mathematical operations, the keys generated, etc. The results are presented clearly and understandably to facilitate analysis and subsequent use.

3) Options of configuration

The main interface also offers configuration options, allowing the user to customize certain application settings. It may include selection encoding, etc. The configuration options are accessible from a scrolling menu.

The main interface of the ECC application can provide a user experience smooth and user-friendly, allowing the user to easily navigate between different features and to carry out of the operations in all simplicity.

4.2. Generation of the Points of an Elliptic Curve

Generating the points of an elliptic curve is a fundamental step in implementing a curve-based cryptography system called Ellipticals. This operation implied the three selections of an appropriate elliptic curve and thus the creation of a base point (also called a generator) on this curve.

Generation of the First Numbers

The generation of the first numbers is essential to guaranteeing the security of cryptographic operations based on elliptic curves. This operation consists of generating large prime numbers, which serve as settings for the construction of the elliptic curves. The next interface user will generate the first numbers with our software.

4.3. Multiplication Scalar

The multiplication scalar is an operation key in cryptography based on elliptic curves. It allows calculating the product of a basis point with a scalar, which is used in many cryptographic algorithms such as encryption and signature.

4.4. Doubly of a Point

Doubling a point is an important operation in calculations on elliptic curves. It consists of calculating the point obtained by multiplying a given point on the elliptic curves. This operation is used in the process of scalar multiplication and other cryptographic algorithms.

1) Addition of two points

The bill of two points is an operation crucial for combine information coming from two points on the elliptic curves. This operation is used in various applications, including encryption, signing and the resolution of the equation Q = kP.

2) Calculation of Degree of a Point and belonging of a point has the curve

Calculating the degree of a point on an elliptic curve is an operation that determines how many times a point can be added to itself before obtaining the point at infinity. This calculation is important for various cryptographic applications, especially for determining the order of an elliptic curve.

Calculation of degree of a point: Either a elliptic curve E 524287 (439.17) and P (524202.493646) ϵ E.

3) checking the membership of a point on a curve: In using the curve E (4.2) (0.7), we can check belonging of point:

Gx = 55066263022277343669578718895168534326250603453777594175500187360389116729240;

Gy = 32670510020758816978083085130507043184471273380659243275938904335575337482424.

With the curve obtained here by the operation of generating all the points of an elliptic curve, the proposition of the study case doesn't have the interface dedicated to this kind of operation.

Therefore, Encryption with the ECC is an operational capital to ensure the confidentiality of the data. This operation used the mathematical properties of elliptic curves to encrypt messages using the recipient's public key. And Decryption with ECC is the reverse process of encryption. It allows the authorized recipient to retrieve the original message using their corresponding private key. Using the same curve parameters as the previous example, the program helps clarify the message again.

5. Conclusion

The method proposed in this article uses Elliptic Curve Cryptography, a more profitable standard cryptosystem. ECC is very useful on small devices such as mobile phones because it requires less key size and has scheduling power. To ensure the same level of protection, the present use of ECC is preferable to using a 256-bit RSA key. Several libraries written in Python were presented in this study. Compared to other techniques such as RSA, this research study shows in detail how ECC, which uses Python, provides a level of security that requires fewer CPU resources to encrypt and decrypt data.

Acknowledgements

The authors of this paper would like to thank everyone who contributed to this paper.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Zhang, H. (2014) Cryptography and Computer Security. Master's Thesis, University of Liège, Liège, 2009-2010.

[2] Joye, M. (2002) Elliptical Cryptography and Smart Card Security Cryptology, **5**, 81-110.

[3] Blake, I.F., Seroussi, G. and Smart, N.P. (1999) Elliptical Curves in Cryptography. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9781107360211

[4] Rayarikar, R., Upadhyay, S. and Pimpale, P. (2012) SMS Encryption Using AES Algorithm on Android, *International Journal of Computer Applications*, **50**, 12-17. https://doi.org/10.5120/7909-1038

[5] Menezes, A.J., Van Oorschot, P.C. and Vanstone, S.A. (1999) Manual of cryptography.

[6] Rao, O.S. and Setty, S.P. (2011) Comparative Study of Arithmetic and Huffman Compression Techniques for Enhancing Security and Effective Bandwidth Utilization in the Context of ECC for Text. *International Journal of Computer Applications*, **29**, 44-60. https://doi.org/10.5120/3566-4905

[7] Koblitz, N. and Menezes, A. (2013) Elliptic Curve Cryptography: The Snake in the Garden. IACR Cryptology ePrint Archive. 2.

[8] Gupta, S., Vashisht, S., Singh, D. and kushwaha, P. (2019) Enhancing Big Data Security using Elliptic Curve Cryptography. 2019 *International Conference on Automation, Computational and Technology Management* (*ICACTM*), London, 24-26 April 2019, 348-351. https://doi.org/10.1109/ICACTM.2019.8776764

[9] Washington, L.C. (2008) Elliptic Curves: Number Theory and Cryptography. CRC Press, Boca Raton.

[10] Menezes, A.J., van Oorschot, P.C. and Vanstone, S.A. (1996) Handbook of Applied Cryptography. CRC Press, Boca Raton, Chapter 13.

[11] Joye, M. (2000) Cryptography on Elliptic Curves. *Techniques and Computer Science*, **19**, 1135-1164.

[12] Vercauteren, F. (2004) Cryptography Based on Elliptic Curves. *Review International of Geometry*, **14**, 71-93.

[13] Giraud, C. and Tillish, J.P. (2006) Curve-Based Cryptography Elliptical Trainers Techniques the Engineer. Ref. J2511.

[14] Cohen, H., Doche, C. and Frey, G.G. (2010) Cryptography on Elliptical Curves. *Cryptography and Network Security*, Hermes Science Publications, Paris, 305-328.

[15] Joye, M. and Quisquater, J.J. (2001) Modern Cryptography: Lessons and Exercises Corrected. Springer, Paris.

[16] Lange, T. and Véron, P. (2009) Introduction to Cryptography: Course and Corrected Exercises. Dunod, Paris.

[17] Buchmann, J. and Walter, M. (2003) Introduction to Cryptography. Springer Berlin Heidelberg, Germany.

[18] Verheul, E.R. and Cohen, H. (2000) Cryptography: Theory and Practice. Hermes Science Publications, Paris.

[19] Thayer, W. (2014) Benefits of Elliptic Curve Cryptography. https://pkic.org/2014/06/10/benefits-of-elliptic-curve-cryptography/