Journal of Autonomous Intelligence (2023) Volume 6 Issue 3
doi: 10.32629/jai.v6i3.740

ORIGINAL RESEARCH ARTICLE

Design and implementation of secured file delivery protocol using enhanced elliptic curve cryptography for class I and class II transactions

Smitha Sasi[1], Srividya Bharadwaj Venkata Subbu[1], Premkumar Manoharan[2,*], Laith Abualigah[3,4,5,6,7,8,9]

[1] *Department of Electronics and Telecommunication Engineering, Dayananda Sagar College of Engineering, Bengaluru 560078, Karnataka, India*

[2] *Department of Electrical and Electronics Engineering, Dayananda Sagar College of Engineering, Bengaluru 560078, Karnataka, India*

[3] *Computer Science Department, Prince Hussein Bin Abdullah Faculty for Information Technology, Al al-Bayt University, Mafraq 25113, Jordan*

[4] *Department of Electrical and Computer Engineering, Lebanese American University, Byblos 13-5053, Lebanon*

[5] *Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan*

[6] *MEU Research Unit, Middle East University, Amman 11831, Jordan*

[7] *Applied Science Research Center, Applied Science Private University, Amman 11931, Jordan*

[8] *School of Computer Sciences, Universiti Sains Malaysia, Pulau Pinang 11800, Malaysia*

[9] *School of Engineering and Technology, Sunway University Malaysia, Petaling Jaya 27500, Malaysia*

*** Corresponding author:** Premkumar Manoharan, mprem.me@gmail.com

ABSTRACT

This research study introduces an ID-based identity authentication protocol that utilizes the enhanced elliptic curve digital signature algorithm, a cryptographic method developed on elliptic curve cryptography. The protocol enhances the Consultative Committee for Space Data Systems (CCSDS) File Delivery Protocol (CFDP), a pioneering protocol explicitly defined for distant space communications. This study employs both dependable and uncertain modes of the CFDP protocol. To make more secure data transactions, two key security risks are effectively mitigated in this research as a result of applying the proposed enhanced elliptic curve cryptography algorithm (ECC) over the ternary galois field. First, it thwarts the impersonation of a harmful entity during a passive attack. Second, it prevents masquerade attacks, further reinforcing the security of space data transmission. This ID-based authentication protocol, therefore, offers a significant advancement in protecting far-space communications, optimizing the integrity of data exchanged across vast distances.

Keywords: authentication protocol; consultative committee for space data systems; CCSDS file delivery protocol; elliptic curve cryptography

ARTICLE INFO

Received: 15 June 2023
Accepted: 26 July 2023
Available online: 6 September 2023

1. Introduction

As the frequency of interplanetary missions continues to rise and scientific space objectives become more complex, the importance of onboard data utilization has become increasingly pronounced. Hence, the spacecraft onboard computer (OBC), which controls mission fulfilment, has become highly refined in hardware and software. Additionally, the ground control center serves as a lone jurisdiction for space regulation and a portal for space-generated actual time data circulation. As the separation increases and the motion of the earth and other planets, we cannot expect the transmission channel from

spacecraft to the ground mission control center (MCC) to be everlasting. During these situations, it is required to design new approaches for file transfers to deliver space missions. To acknowledge the prevailing demands, CCSDS introduces the CCSDS File Delivery Protocol (CFDP), a novel file transfer protocol designed for both space-based and ground-based networks[1]. Traditional authentication schemes rely on symmetric algorithms, such as secure hash algorithms, which have long been considered reliable and necessitate secret keys. Nonetheless, handling and safeguarding the secret keys is a tough task. Thus, to address the coordination problem, Elliptic Curve Cryptography (ECC) is utilized as a viable solution. ECC involves using a pair of cryptographic keys—a private key and a corresponding public key—assigned to each device. When sending a message, the sender signs it using their private key, while the recipient can verify the signature's authenticity by employing the sender's public key[2]. If the message is altered before reaching the receiver, the signature authentication is declined as the authentic signature is invalid for the altered message. This study aims to discuss a particular implementation of the CFDP with the ECDSA and to show its practical usage. **Figure 1** shows the layered architecture of security implementation on CFDP protocol.

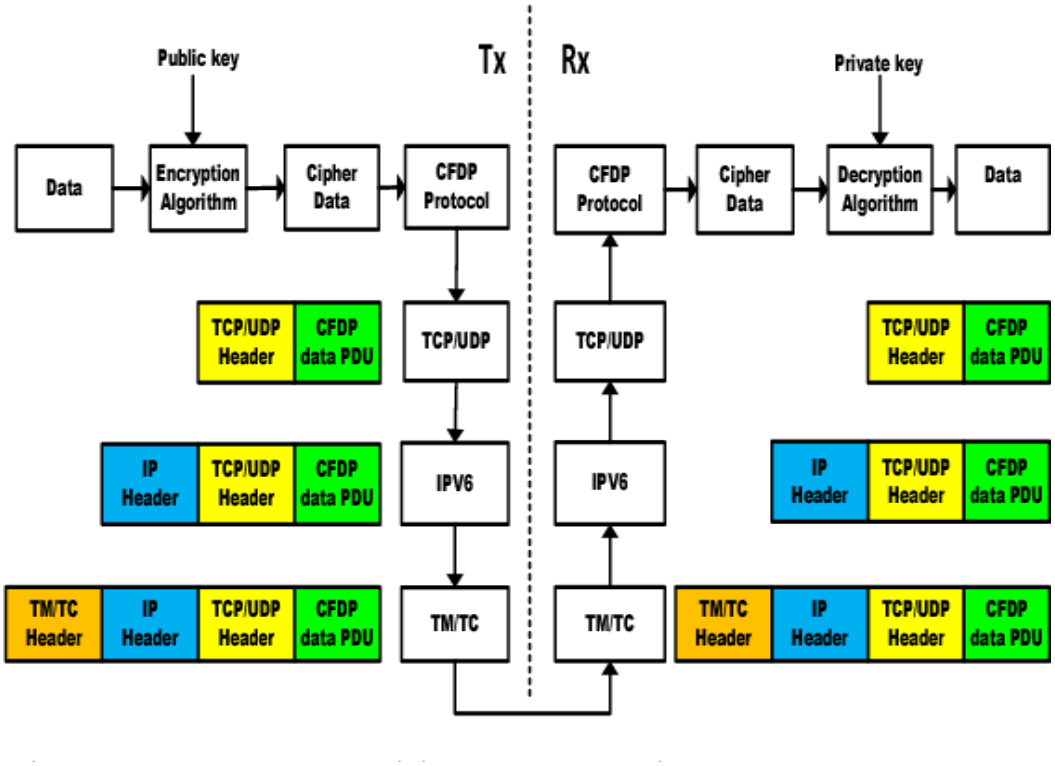

Figure 1. Implementation of the CFDP.

The existing space data communication uses RSA for confidentiality and authentication of data. RSA and ECC are the most productive techniques among all asymmetric encryption algorithms[3]. They have a huge number of benefits in contrast with different cryptosystems. RSA is a principal public-key cryptosystem generally utilized for secure data transmission. In such a cryptosystem, the encryption key is public and varies from the decryption key is kept secret. In RSA, this asymmetry depends on the functional trouble of factoring the result of two substantial prime numbers, the factoring issue[4]. RSA represents Ron Rivest, Adi Shamir and Leonard Adleman, who first publicly depicted the calculation in 1977. A client of RSA makes and distributes a public key dependent on the two extensive prime numbers that must be kept secret. Anybody can use the public key to scramble a message; however, with distributed strategies, if the public key is sufficiently huge, just somebody with information on the prime numbers can decode the message[5]. The limitations of RSA stem from its requirement of a large number of bits, which can pose challenges when dealing with computationally intensive tasks, such as encoding large volumes of data on a single computer. Additionally, RSA relies on third-party verification to ensure the reliability of public keys, which introduces potential vulnerabilities. The

use of RSA encryption also leaves the exchanged data susceptible to manipulation by intermediaries who may tamper with the public key infrastructure[6].

Elliptical curve cryptography is a strategy for encoding data records with the goal that just explicit people can decode them. ECC depends on the arithmetic of elliptic curves and uses the area of points on an elliptic curve to encode and unscramble data. ECC bears productive execution of wireless security highlights, for example, secure electronic mail and Web perusing. A huge preferred standpoint of EC over RSA is a considerably shorter key (and data length because of cushioning). Beginning to deal with 4096-bit keys (512-bytes) is getting difficult. Identical security of a 3072 RSA key can be accomplished with a 256-bit EC key[7]. ECC is an extremely promising and new field to locate a more cost-effective strategy to perform encryption for versatile gadgets and verify image transmission over the web. Elliptic curves are accepted to give great security little key sizes, which are exceptionally valuable in numerous applications. Littler key sizes may result in quicker execution timings for the plans, which is gainful to frameworks where real-time execution is a basic factor[8]. We have appraisals of parameter sizes giving equal security dimensions to RSA and ECC frameworks[9]. These correlations show the intrigue of elliptic curve cryptography, particularly for applications that have high security. The use of ECC is exceedingly prescribed to make greater security and higher speed without expanding the computational burden. Then again, with the expansion of smaller embedded gadgets designed with additional confinements (for example, Calculation Power, Memory and Battery life) and cryptographic plans, particularly in asset-compelled gadgets, should be secure, viable, and less expensive[10]. ECC has littler cost proportion. Besides, to boost the execution of the recently designed gadgets, ECC itself needs predictable improvement[11].

Several authors have conducted a comparative analysis between RSA and ECC in the literature, considering various measurement parameters, including security performance. One study compared the point multiplication operation of an elliptic curve in RSA and ECC on two 8-bit processor computer systems. The results indicated that ECC-160-point multiplication demonstrated greater efficiency than RSA-1024 private key operation[12]. The risk associated with key usage depends on the key length of RSA and ECC. The authors concluded that until 2014, a 1024-bit RSA key presented a small level of risk, whereas a 160-bit ECC key over a prime field could be safely used for an extended period[13]. While RSA exhibits faster performance compared to ECC, it is important to note that ECC surpasses RSA in terms of security. The utilization of digital signatures in RSA and ECC suggests that RSA could be a suitable option for applications that prioritize message verification over the generation of the signature itself. According to research findings[13], ECC surpasses RSA in both operational efficiency and security. While RSA is widely recognized as the first practical asymmetric-key cryptosystem and has become the de facto standard for public-key cryptography, its security relies on the difficulty of integer factorization. However, RSA's decryption process is less efficient than its encryption process. To compare RSA and ECC, encryption and decryption were performed on three sample input data sets of 8 bits, 64 bits, and 256 bits using random keys following NIST recommendations. The experimental results revealed that ECC outperforms RSA regarding operational efficiency and security, even with fewer parameters. ECC is particularly suitable for devices with limited resources[14]. The primary objective of this study is as follows.

- To study existing cryptographic methods for space data communication.
- To study CCSDS File Delivery Protocol for NCC to satellite for core procedures.
- To design and implement enhanced ECC for transaction.

The paper is organized as follows. Section 2 discusses the design concept and the elements of the CCSDS file delivery protocol. Section 3 briefly discusses the CCSDS file delivery protocol and ECC operation. Sections 4 and 5 present the digital signature and elliptic digital signature algorithm. Section 6 presents the operation of the enhanced elliptic curve cryptography, and Section 7 comprehensively discusses the results obtained from various case studies. And Section 8 concludes the paper.

2. Design concept and architectural elements of CFDP

The protocol contains core procedures and extended procedures. This research work proposed Class I and Class II communications. Class I is an unreliable mode of core procedures. Class II is a reliable mode of core procedures. The core procedure supports point-to-point communications, and the extended procedure addresses point-to-multipoint communications. The former communicates amidst protocol entities with an explicit network pathway[15]. The source is the one from which the file is copied to perform a file copy exercise, which is copied to the receiver. In case the sender and the receiver are not able to be connected directly by the network, the extended procedures accordingly conform to an end-to-end file copy activity by carrying out several file copy activities, single file copy activity with source and first waypoint; others constituting subsequent waypoints; and ultimate file copy procedure with the final waypoint and target node. These are the cases of core file copy procedures. For a transaction to be reliable, it must choose whether to conduct in acknowledged or unacknowledged modes. If an unacknowledged mode is chosen, data distribution failing will not be announced to the source; hence, reconstruction cannot be performed even if errors are present and such data is cast off. Hence it cannot be assured that the entire file is received. When using the acknowledged mode, the sender is notified by the receiver of data that is not delivered and transmitted again, ensuring exhaustive data delivery[16].

The structural components are illustrated in **Figure 2**. It is demonstrated that every protocol entity can access just a specific filestore and a specific user.

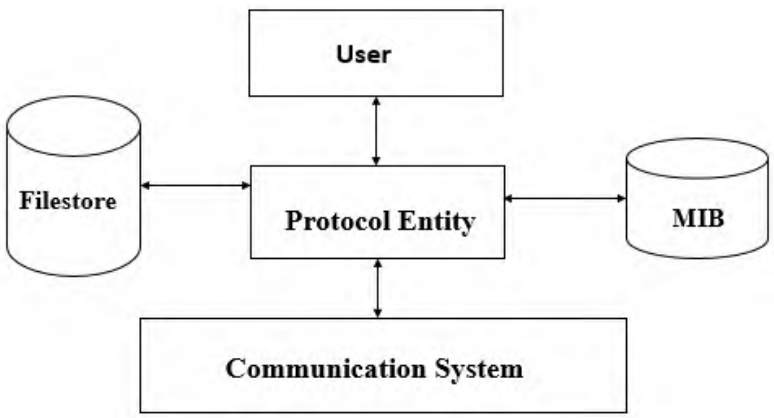

Figure 2. Units of file delivery protocol.

When the CFDP user makes a request, the protocol is initiated. The interaction of the user with the protocol is through service primitives. A CFDP user is a software undertaking that can be exerted manually. There is exactly one user assigned to the CFDP entity. In case there is no user, extended procedures waypoint comes into effect. Protocol entity comprises core delivery operations, which grant prompt file delivery and control across an individual network node, allowing for separate time or prompt transmission across several nodes with the right competence for forward routing. The protocol is operated by replicating files from different storage media, which means that all the entities have a local copy[17]. As there are several different storage facilities, the protocol comprehends that a filestore be represented as a standard representation termed as 'Virtual File store', is designated a definitive group of attributes, and the protocol maintains the file transmission procedure using them. Hence in this way, entire freedom is provided by the technology that uses the file store. Static data between local users and local and remote CFDP entities are managed as system tables or Management Information Base (MIB), which has default values for user communication requirements. The underlying communication system protocol considers the opportunity for transmission framework to be available to which the CFDP entities have access. The protocol services are not complicated because they

might have to conduct over an endless scope of applications. This elemental theoretical delivery system is called the Unit data Transfer (UT) layer. As just the basic network facilities are considered, occasionally, CFDP contributes to the services administered by the UT layer[18].

3. CFDP protocol operations and elliptic curves

CFDP contains core file-delivery and extended file-delivery operations. The former provides elemental file delivery operations between two points realized over an individual link. The latter is devised for point-to-point data transmission in increasingly complicated assignments where in the absence of a direct channel from source to destination, it facilitates multi-hop transmission over an erratic network with numerous channels. The CFDP protocol renders QoS for both unreliable and reliable modes. Within the context of ECC, the Digital Signature Standard (DSS) categorizes elliptic curves into two types: pseudo-random curves and special curves. Pseudo-random curves derive their coefficients as an output from the cryptographic hash function. On the other hand, special curves enhance the performance of elliptic curve operations by incorporating upgraded coefficients and hidden fields. Pseudo-random curves may be exemplified across prime fields $GF(p)$ and also binary fields $GF(2^m)$[19].

This research focuses on ECC $GF(3^m)$. A prime field $GF(p)$ consists of a prime number p of elements. The field consists of integers modulo p. The elliptic curve has the form $y^2 = x^3 + ax + b$. **Figure 3** is an example of a prime field modulo 23.

The geometry of elliptic curves is elaborated on in this study. The steps in the geometry of elliptic curves are as follows.

Adding points on an Elliptic curve: Consider the elliptic curve E, $y^2 = x^3 - 5x + 8$. To add points to the curve, consider points P and Q. Draw a line L across the selected points. The curve is intersected by the line at a third point called R. A vertical line must be drawn through R to intersect the curve at E[20]. The sum of P and Q on the curve E is defined as a reflected point and denoted as P $\oplus$ Q or P + Q. The adding points on the elliptic curve are visualized in **Figure 4**.

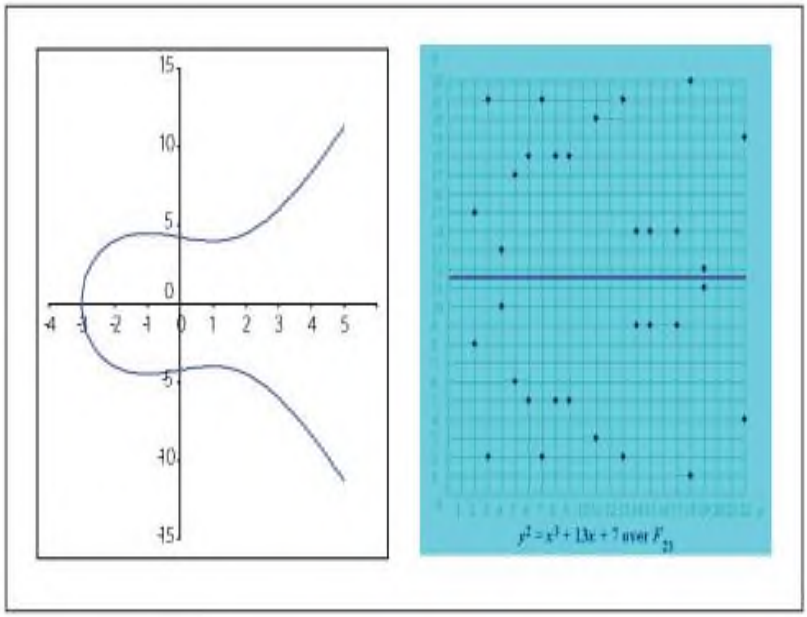

Figure 3. 3rd-degree elliptic curves, real domain (left), over the prime field (right).

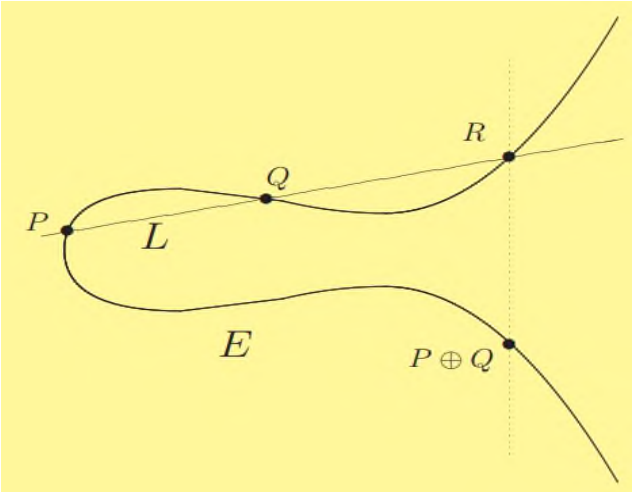

Figure 4. Adding two points, P and Q, and drawing a line L across the third point.

Adding a curve point to itself: Considering the same Elliptic curve, the main task is to add the point to itself, while several lines have to go through the same point[20]. If P has to be added to Q, and Q approaches P, then the line L is tangential to the curve at the point P. Now a third intersecting point is considered, reflected along the x-axis and culminates as P ⊕ P or 2P.

Vertical lines and the "extra point" at infinity: Consider point P on the curve in **Figure 5** and its reflected point P[21]. The issue that arises is that a line through P and −P do not converge to give another point. As a solution, an extra point O, which is a point on every vertical line, is considered out of the plane, supposedly at infinity[22].

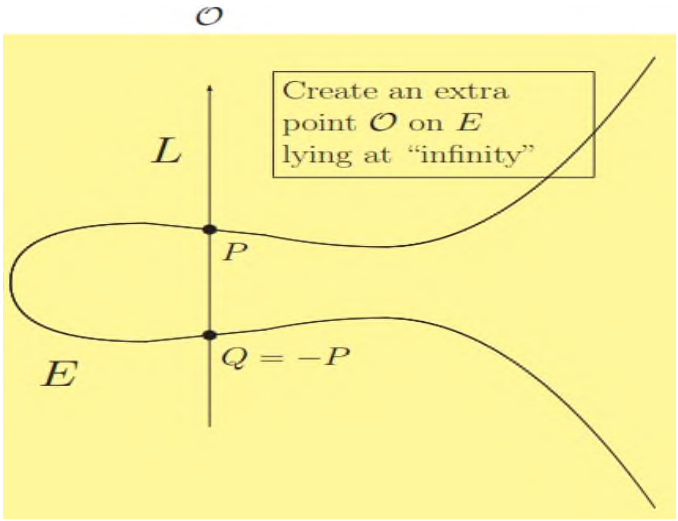

Figure 5. Vertical line across two points having no third intersection point with E and creation of point O at infinity.

4. Digital signature

Digital signatures are based on public-key and used for message verification. In the actual physical world scenario, handwritten signatures or names are written for personal identification and verification of the message's origin[23]. In a similar vein, a digital signature serves as a method to securely bind the digital information of an individual or entity in a way that can be verified by both the recipient and any third party involved. A digital signature is a cryptographic value obtained from the data and a secret key which is secret to the signer. In the physical scenario, the receiver of the message has to be sure of the origin and the sender and should not be able to reject it. This is very important in business applications where sensitive data is

exchanged, and disagreements should not be over the information exchanged[24]. Digital signature has its basis in public key cryptography, and its design is interpreted as shown in **Figure 6**[25].

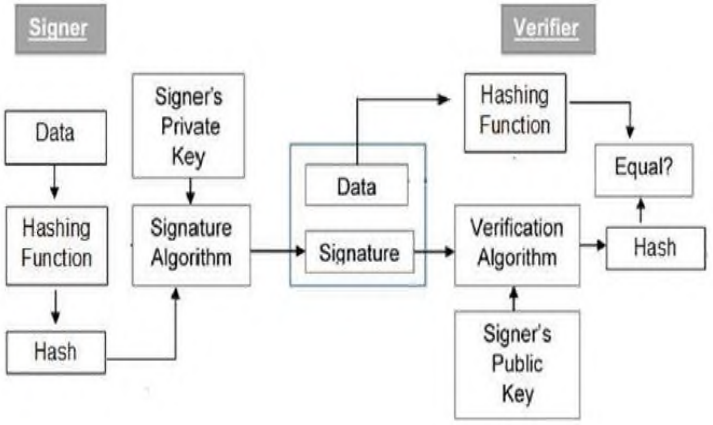

Figure 6. Digital signature model.

Every individual implementing this design owns a pair of public-private keys. Normally, the key pairs utilized for enciphering/deciphering and signing/verifying are different. The private key (signature key) is used for approval, and the public key verifies. The signer provides the data to the hash function and obtains the hash. Hash value and signature key are input to a signature algorithm to form a digital signature. The data is combined with the signature and communicated to the verifier. The digital signature and the verification key are input to the verification algorithm. Verifier implements the hash function on obtained data to produce a hash value. To verify, the hash value and output of the verification algorithm are set side by side, thereby deciding if the digital signature is valid. As a digital signature is formed by the 'private' key of the signer, the signer cannot commit repudiation.

Considering the diverse range of digital communications, it is required that encrypted data is transmitted instead of plaintext to support privacy. In the public key encryption model, the sender's public key can be obtained, and encrypted messages can be sent by spoofing identity[26]. As a result, users utilize Public Key Cryptography (PKC) to achieve digital signatures alongside encrypted data, ensuring message authentication and non-repudiation. This is accomplished by combining digital signatures with encryption schemes. Let us, for a short time, discuss how to attain this requirement. A digital signature is either done by the sign then encrypt method or encrypt then sign method. The receiver can use the sign-then-encrypt crypto model to spoof the sender's identity and send data to another party, because of which this scheme is not adopted. The procedure of encrypt-then-sign is staunch, broadly accepted, and illustrated in **Figure 7**.

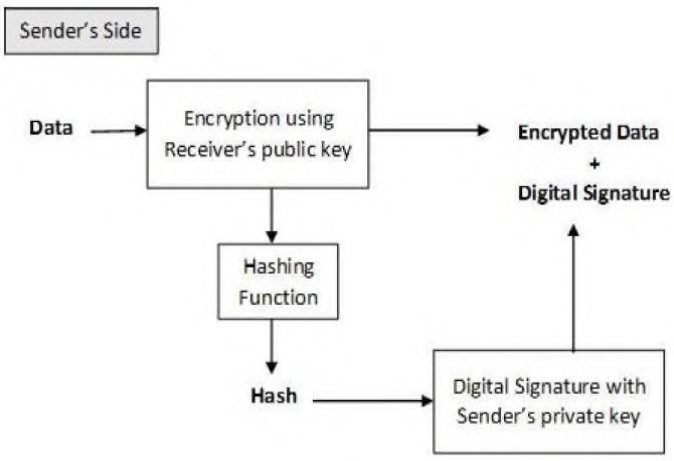

Figure 7. Encryption and digital signature.

5. Elliptic curve digital signature algorithm

ECDSA, or the Elliptic Curve Digital Signature Algorithm, interprets the widely-used DSA algorithm to obtain security levels with smaller key sizes. This is done using elliptic curve cryptography, based on PKC, advanced in the mid-2000 s. Instead of depending on a large integer which is a product of several large prime factors, it uses an ECDLP (elliptic curve discrete logarithmic problem)[9,10]. As ECDLP is considerably more complicated than DLP, the strength-per-key-bit is significantly higher in elliptic curve systems than in traditional discrete logarithm systems. Hence, lower value elements are used in ECC than with DL (Discrete Logarithm) systems with comparable security levels. Smaller parameters provide leverage, such as speed and shorter keys and certificates. They have crucial usage in constrained environments. The ECDSA authenticator should learn the private key to operate. The public key is obtained from the private and domain parameters. They are stored in the authenticator's repository. The private key is not available to everyone. But the public key should be available to everyone. **Figure 8** shows the key generation process.

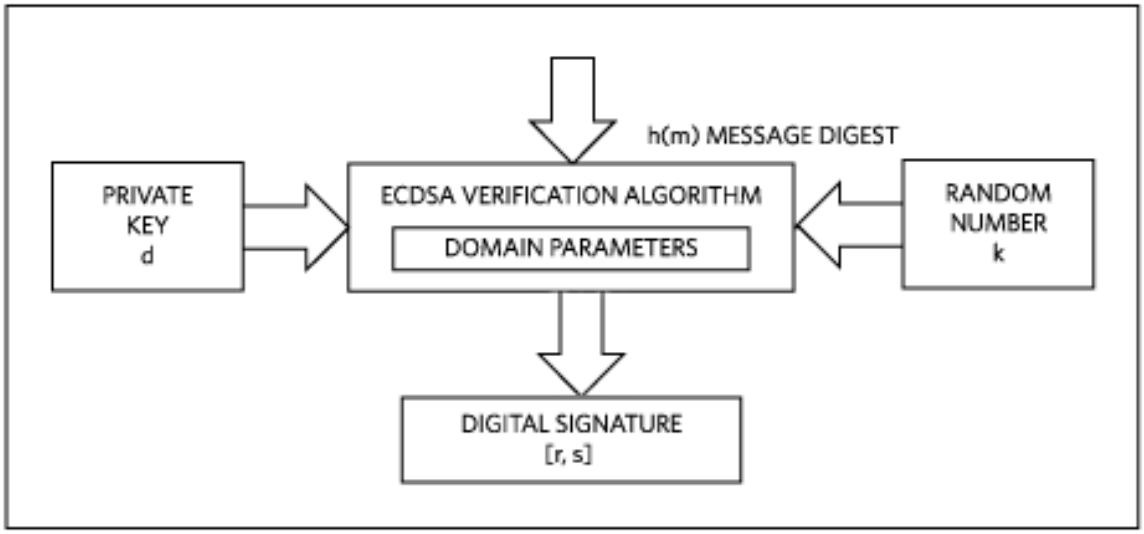

Figure 8. Process of generating key pair.

The random number generator obtains private key d (a scalar). The public key $Q(x, y)$ is calculated as Equatioon (1).

$$(x, y) = d \times G(x, y) \tag{1}$$

A digital signature plays a crucial role in verifying authenticity by utilizing the public key of the authenticator. To begin with, the original message of variable length is transformed into a fixed-length message digest, denoted as *h*(*m*), by applying a secure hash algorithm. This algorithm possesses specific characteristics:

1) irreversibility, making it impossible to retrieve the original message from its digest through any calculations;

2) collision resistance, ensuring it is highly unlikely to obtain multiple messages that result in the same digest; and 3) high avalanche effect, where even a slight modification in the message results in a significant alteration in the digest. Once the message digest is obtained, a random number generator generates a value of '*k*' for subsequent curve calculations. **Figure 8** depicts the mechanism.

The signature comprises *r* and *s*, which are both integers. Equation (2) depicts how *r* is obtained from *k*, which is a random number and base point $G(x, y)$:

$$\left.\begin{array}{c}(x_1, y_1) = k \times G(x, y) mod p \\ r = x_1 \bmod n\end{array}\right\} \tag{2}$$

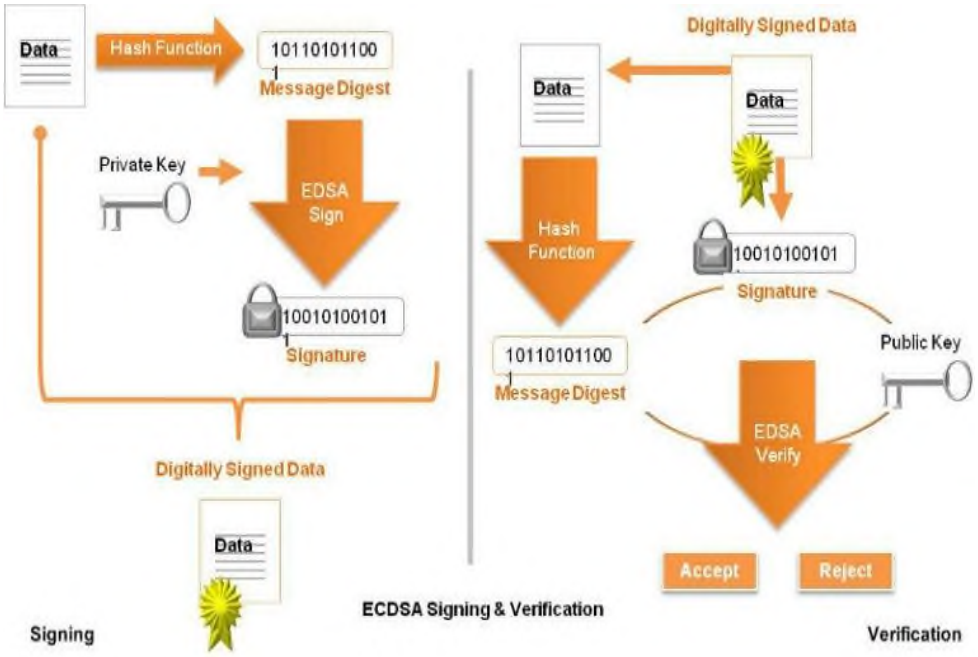

Figure 9. Process of computation of the signature and verification.

To prove validity, *r* must be anything other than zero. If *r* is 0, a different random number, *k*, is obtained, and *r* is calculated afresh. Now, *s* is calculated as per Equation (3). The message digest $h(m)$; the private key d; r; and the random number k are used as inputs:

$$s = (k^{-1}h\big((m) + d \times r\big)mod\, n)) \tag{3}$$

To prove validity *s* must be anything but zero. In case *s* is 0, *k*, which is a random number, is obtained newly, and values of *r* and *s* are calculated again.

Signature verification and signature computation are correlated. It serves for message authenticity verification by utilizing the public key of the authenticator. The authenticator obtains and signs a message digest using the secure hash algorithm for signature computation.

Public keys *Q(x, y)*, r and s are used to obtain the result. **Figure 9** depicts the mechanism of signature and verification. Equation (4) presents separately how the verification process is carried out. The essential components involved in the digital signature process include the message digest $h(m)$, the public key $Q(x, y)$, as well as the signature components r and s. These elements, which are given as inputs, work in conjunction with the base point $G(x, y)$ of the elliptic curve.

$$\left.\begin{array}{c} w = s^{-1}modn \\ u_1 = (h(m) \times w)modn \\ u_2 = (r \times w)modn \\ (x_2, y_2) = \big(u_1 \times G(x, y) + u_2 \times Q(x, y)\big)modn \end{array}\right\} \tag{4}$$

For the verification to be successful, x_2 has to be equal to r, which tells the signature was obtained by using the private key[9].

6. Enhanced elliptic curve cryptography

In this research would like to emphasize that the uniqueness of this proposal is the representation of the elements of Galois Field in its p's ternary form $GF(3^m)$ to ensure enhanced security of the information. As an illustration, let us consider $GF(3^3)$. The elements of $GF(3^m)$ are constructed using the primitive polynomial $p(x) = x^3 + 2x^2 + 1$.

Let α be a root of the polynomial *p(x)*.

$$p(x = \alpha) = \alpha^3 + 2\alpha^2 + 1 = 0$$
$$\alpha^3 = 2\alpha^2 + 1$$

Hence $\alpha^3 = \alpha^2 + 2$.

Table 1. The elements of $GF(3^3)$ constructed using the primitive polynomial $p(x) = x^3 + 2x^2 + 1$.

Elements	Polynomial representation	Ternary and decimal representation	3's complement representation
0	0	$(000)_3 = (0)$	001
1	1	$(100)_3 = (1)$	201
α	α	$(010)_3 = (3)$	221
α^2	α^2	$(001)_3 = (9)$	000
α^3	$2 + \alpha^2$	$(201)_3 = (11)$	100
α^4	$2 + 2\alpha + \alpha^2$	$(221)_3 = (17)$	010
α^5	$2 + 2\alpha$	$(220)_3 = (8)$	011
α^6	$2\alpha + 2\alpha^2$	$(022)_3 = (24)$	202
α^7	$1 + \alpha^2$	$(101)_3 = (10)$	200
α^8	$2 + \alpha + \alpha^2$	$(211)_3 = (14)$	020
α^9	$2 + 2\alpha + 2\alpha^2$	$(222)_3 = (26)$	002
α^{10}	$1 + 2\alpha + \alpha^2$	$(121)_3 = (16)$	110
α^{11}	$2 + \alpha$	$(210)_3 = (5)$	021
α^{12}	$2\alpha + \alpha^2$	$(021)_3 = (15)$	210
α^{13}	2	$(200)_3 = (2)$	101
α^{14}	2α	$(020)_3 = (6)$	211
α^{15}	$2\alpha^2$	$(002)_3 = (18)$	222
α^{16}	$1 + 2\alpha^2$	$(102)_3 = (19)$	122
α^{17}	$1 + \alpha + 2\alpha^2$	$(112)_3 = (22)$	112
α^{18}	$1 + \alpha$	$(110)_3 = (4)$	121
α^{19}	$\alpha + \alpha^2$	$(011)_3 = (12)$	212
α^{20}	$2 + 2\alpha^2$	$(202)_3 = (20)$	022
α^{21}	$1 + 2\alpha + 2\alpha^2$	$(122)_3 = (25)$	102
α^{22}	$1 + \alpha + \alpha^2$	$(111)_3 = (13)$	120
α^{23}	$2 + \alpha + 2\alpha^2$	$(212)_3 = (23)$	012
α^{24}	$1 + 2\alpha$	$(120)_3 = (7)$	111
α^{25}	$\alpha + 2\alpha^2$	$(012)_3 = (21)$	212
α^{26}	1	$(100)_3 = (1)$	001

As can be seen, **Table 1** shows the elements of $GF(3^3)$ constructed using the primitive polynomial $p(x) = x^3 + 2x^2 + 1$. The polynomial representation, the ternary representation, and the decimal representation of the galois field elements are depicted in **Table 1**. This ternary representation of the data is used for all the mathematical operations like addition, multiplication, inverse and all the other relevant modular operations in the proposed algorithm. In this research, the ternary representation of ECC over galois field is utilized for representing the following:

- The information that needs to be secured.
- The mathematical relation between the private key and public key in customized ECC.

- The modular operations use an Elliptic digital signature algorithm for authentication.

Figure 10 explains the Elliptic curve cryptography's random key generation and implementation over the ternary galois field. Simple implementation of ECC over finite field $GF(3^3)$ and $GF(3^{-3})$ system setup.

ECC over finite field $GF(3^3)$

The underlying finite field will be F_3 m, and using the generator point $p = \alpha^3, \alpha^5$ which has order $E(F_3 m) = n = 27$;

Step 1: Key generation

Entity A performs the following operation

A selects a random integer $K_A = 3$ from$[1, n-1]$

A computes the point $A = K_A \times P = 3 \times (\alpha^3, \alpha^5) = \alpha^{12}, \alpha^{18}$

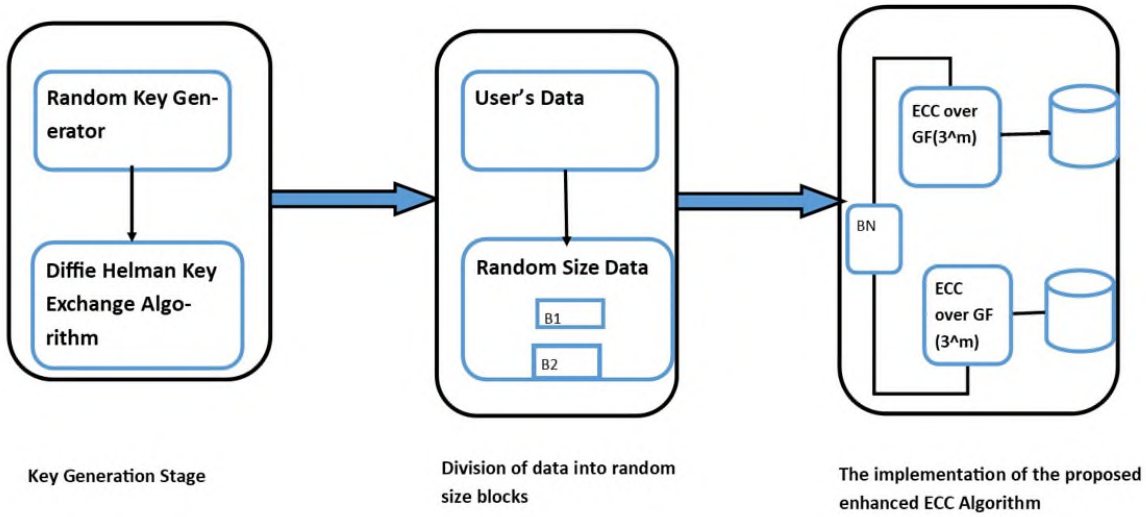

Figure 10. Process of computation of the ECC over ternary galis field.

Step 2: Encryption process

Entity B Sends Message to A

Message $M = (10100000100) = \{(2203)_3, (020)_3)\} = (\alpha^5, \alpha^{14})$

Entity B performs the following steps.

Look up the public key of A from the public key sensor: $A = \alpha^{12}, \alpha^{18}$

Represent M as a pair of elements $(M1, M2)$

$M1 = (101000), (00100) = (220), (020)$ over $GF(3^m)$

Selects the random integer $K_B = 2$ from $[1, n-1]$

Computes his public key: $B = K_B \times P = 2 \times (\alpha^3, \alpha^5) = (\alpha^{18}, \alpha^{24})$

Computes the shared Key = $SBA = K_B \times A = 2 \times (\alpha^{12}, \alpha^{18})$

$(x3, y3) = (\alpha^{20}, \alpha^{22})$

The shared key SBA is treated as the Session Key. Computes the field element by considering the session key and raising it to the order of the Galois field m.

$(x4, y4) = ((SBA(x))^M, ((SBA(y))^M$

$(x4, y4) = ((\alpha^{(20)^3}, (\alpha^{(22)^3})$over $GF(3^3)$

$= (\alpha^8, \alpha^{14})$

B foams the cipher text

$c1 = ((m1 + x3)(x4)) = \alpha^3$

$C2 = (m2 + y3)(y4) = \alpha^{21}$

B transmits the public key of A and the cipher text to A; $(\alpha^{12}, \alpha^{18}, \alpha^{3}, \alpha^{21})$

Step 3: Decryption process (A)

Entity A decrypts the cipher $(\alpha^{12}, \alpha^{18}, \alpha^{3}, \alpha^{21})$ received from B by performing the following steps:

A computes the session key

$SAB = K_A(\mathrm{B}) = (2) \times (\alpha^{18}, \alpha^{24})$

$(x3, y3) = (\alpha^{20}, \alpha^{22})$

A forms $(x4, y4)$ just as B did $x4 = (\alpha^{(20)^3}) = \alpha^{5}$

$y4 = (\alpha^{(22)^3}) = \alpha^{14}$

A recovers the message $(\mathrm{M1}, \mathrm{M2})$ by Computing

$\mathrm{M1} = \frac{C1}{x4} - x3$ over $GF(3^m) = \frac{\alpha^3}{\alpha^8} - \alpha^{20} = \alpha^5$

$\mathrm{M2} = \frac{C2}{y4} - y3$ over $GF(3^m) = \frac{\alpha^{21}}{\alpha^{14}} - \alpha^{22} = \alpha^{14}$

$(\mathrm{M1}, \mathrm{M2}) = (\alpha^5, \alpha^{14}) = (220{,}020) = (101000001000)$

ECC over finite field $GF(3^{\overline{m}})$

The underlying finite field will be $F_3\,\overline{m}$ and using the generator point $p = \alpha^0, \alpha^{19}$ which has order E (F_3 m) = n = 27;

Step 1: Key generation

Entity A performs the following operation

A selects a random integer $K_A = 3$ from $[1, n-1]$

A computes the point A = $K_A \times P = 3 \times \alpha^0, \alpha^{19} = (\alpha^5, \alpha^7)$

Step 2: Encryption process

Entity B Sends Message to A

Message $\mathrm{M} = (10100000100) = \{(2203)_3, (020)_3)\} = (\alpha^5, \alpha^{14})$

Entity B perform th the following steps.

Look up the public key of A from the public key sensor: $\mathrm{A} = (\alpha^5, \alpha^7)$

Represent M as a pair of elements $(\mathrm{M1}, \mathrm{M2})$

$\mathrm{M1} = (101000), (00100) = (220), (020)$ over $GF(3^{\overline{m}}) = (\alpha^5, \alpha^{14})$

$= (\alpha^{19}, \alpha^{8})$ over $GF(3^{\overline{m}})$

Selects the random integer $K_B = 2$ from $[1, n-1]$

Computes his public key: B = $K_B \times P$=$2 \times (\alpha^0, \alpha^{19}) = (\alpha^{17}, \alpha^{17})$

Computes the shared Key =$SBA = K_B \times$ A=$2 \times (\alpha^5, \alpha^7)$

$(x3, y3) = (\alpha^{14}, \alpha^4)$

The shared key SBA is treated as the Session Key. Computes the field element by considering the session key and raising it to the order of the Galois field *m.*

$$(x4, y4) = ((SBA(x))^M, (SBA(y))^M$$

$(x4, y4)$=$((\alpha^{(14)^3}, \left(\alpha^{(4}\right)$ over $GF(3^3) = (\alpha^{16}, \alpha^{12})$

B foams the cipher text

$C1 = ((m1 + x3)(x4)) = \alpha^{18}$

$C2 = (m2 + y3)(y4) = \alpha^{4}$

CB transmits the public key of A and the cipher text to A; $(\alpha^5, \alpha^7, \alpha^{18}, \alpha^2)$

Step 3: Decryption process (A)

Entity A decrypts the cipher $(\alpha^5, \alpha^7, \alpha^{18}, \alpha^2)$ received from B by performing the following steps:

A computes the session key

$SAB = K_A(B) = (2)(\alpha^{18}, \alpha^{24})$

$(x3, y3) = (\alpha^{14}, \alpha^{4})$

A form $(x4, y4)$ just as B did $x4 = (\alpha^{(14)^3})$=α^{16}

$y4 = (\alpha^{(4)^3})$=α^{12}

A recovers the message (M1, M2) by Computing, $\text{M1} = \frac{C1}{x4} - x3$ over $GF(3^{\overline{m}}) = \frac{\alpha^{18}}{\alpha^{16}} - \alpha^{14} = \alpha^{19}$

$\text{M2} = \frac{C2}{y4} - y3$ over $GF(3^{\overline{m}}) = \frac{\alpha^{2}}{\alpha^{12}} - \alpha^{4} = \alpha^{8}$

$(\text{M1}, \text{M2}) = (\alpha^{19}, \alpha^{8})$ over $GF(3^{\overline{m}}) = (101000001000)$

In this approach, point addition over $GF(3^{\overline{m}})$ has been used. **Figure 11** shows the mapping points for plain text and cipher text mapping of Elliptic Curve cryptography over the Ternary Galois Field.

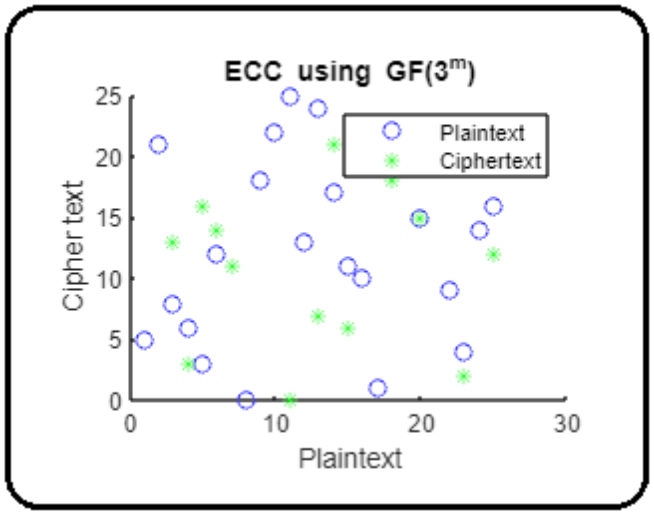

Figure 11. Mapping of points (*x*, *y*) of ECC over ternary galois field.

The comparative time analysis of the enhanced ECC algorithm is shown in **Table 2**. The results proved that elliptic curve cryptography over the ternary galois field consumes less time and is more secure than the conventional elliptic curve method.

Table 2. Comparative time analysis of proposed method with binary galois field ECC.

Time	$\boldsymbol{GF(2^m)}$	$\boldsymbol{GF(3^m)}$	$\boldsymbol{GF(3^{\overline{m}})}$
Time consumption for encryption and decryption process	0.1340 s	0.0938 s	0.1013 s
Key generation time	0.043 s	0.0298 s	0.0298 s
Overall processing time	0.1770 s	0.1236 s	0.1311 s
Throughput	1627 bps	2330 bps	2196 bps

Figure 12 explains the comparison analysis of the proposed method with binary galois field ECC. In the ternary galois field, data transmission for successful packets is more compared to the conventional ECC method.

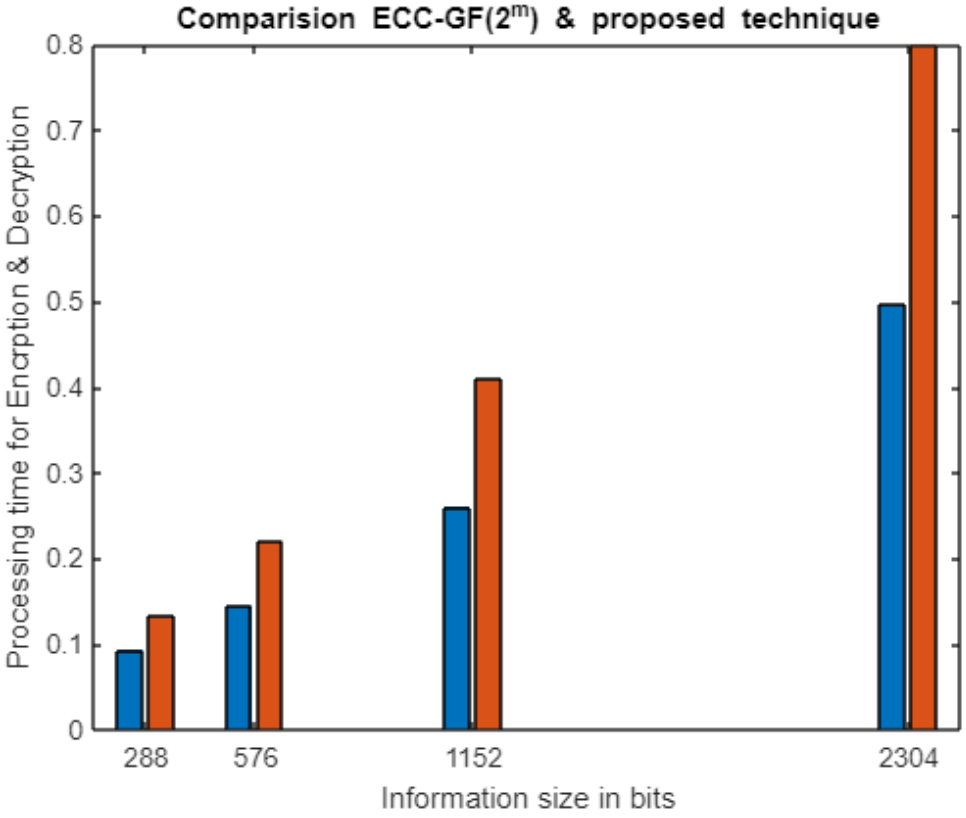

Figure 12. Comparison analysis of the proposed method with binary galois field ECC.

7. Results and discussions

System Configuration: OS-RedHat Linux 7.2
Peer 1. Desktop/Processor-64-Bit/RAM-1GB
Peer 2. Onboard Commuter/Red hat Linux 7.2/Processor-Intel® Core 2 Duo/RAM-1GB
Peer 1 acts as NCC, and Peer 2 is the spacecraft.

The screenshots demonstrate the usage of the CFDP protocol along with the ECDSA over the ternary Galois field algorithm. The code for the protocol is implemented in C language along with socket programming in the UNIX system, and data transmission (peer-to-peer process) is carried out using two UNIX terminals. The code for ECDSA is implemented in Python, and the results are as follows: Firstly, to demonstrate CFDP protocol, the following commands are provided as input to the CFDP user:

Transmission mode: Unreliable transmission mode
Sender
IP Address: 10.0.0.2
Source file name: data.c
Run the command service iptables stop initially.

Figure 13 depicts the user interface by calling put.request service primitive. The transaction is initiated when the user sends a Put request to the CFDP entity. Upon receiving the Put request, the CFDP entity begins the transaction process. It adds the necessary information to the file metadata, encrypts the data using a public key, and starts the file delivery operation. To transmit each item, the CFDP entity organizes it into a Protocol Data Unit (PDU) and transfers it to the lower layer network. The PDU consists of metadata, an encrypted file PDU, and an End of File PDU. During this operation, the CFDP data is transmitted using the IP layer. The CFDP data is attached with an IP header to form an IP datagram, which is then transmitted to the Data Link Layer. The data link layer supports CCSDS path services, which package the PDUs into CCSDS packets, virtual channels, and frames. These are then transmitted over the physical channel to the ground station.

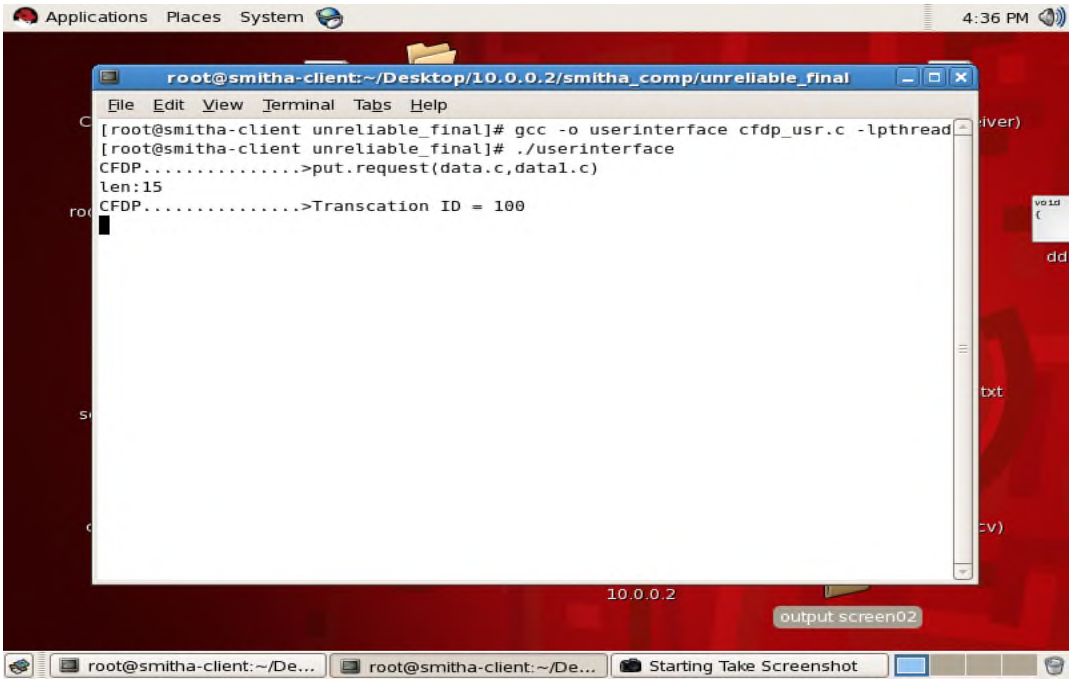

Figure 13. User interface program.

The transmission core program accepts the input data from the user interface program through a pipe and starts transmitting the CFDP data shown in **Figure 14**: The metadata and file PDU transmits to the receiver side.

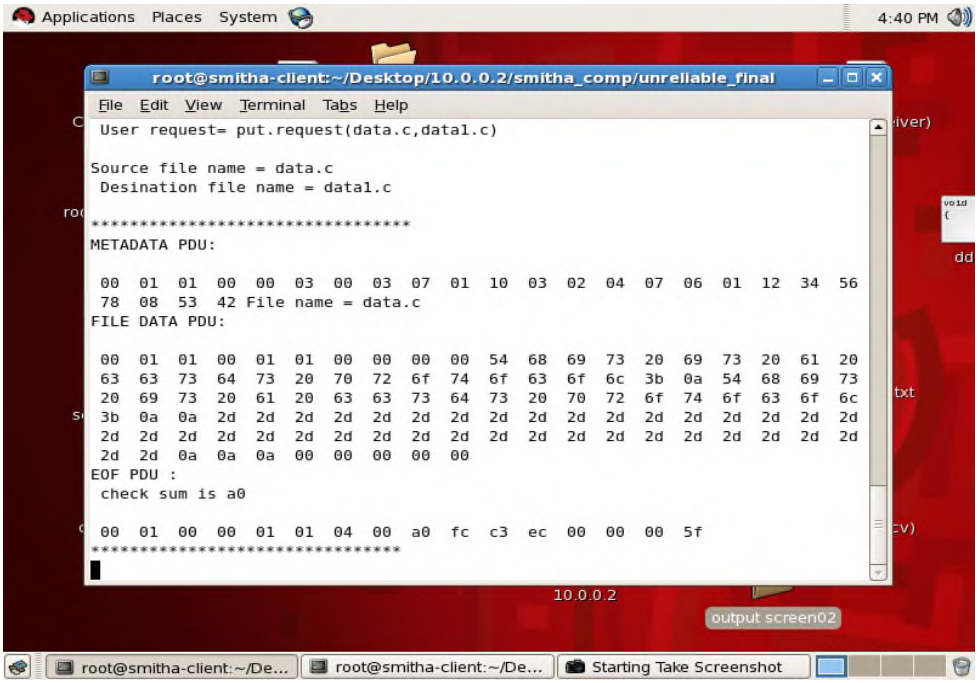

Figure 14. Core procedure output at the sender.

Receiver

IP Address: 10.0.0.1

Figure 15 shows a space image of remote sensing data and encrypted by using ECC obtained from the hex file, as shown in **Figure 16**. The encrypted data is signed by ECDSA, as shown in **Figure 17**.

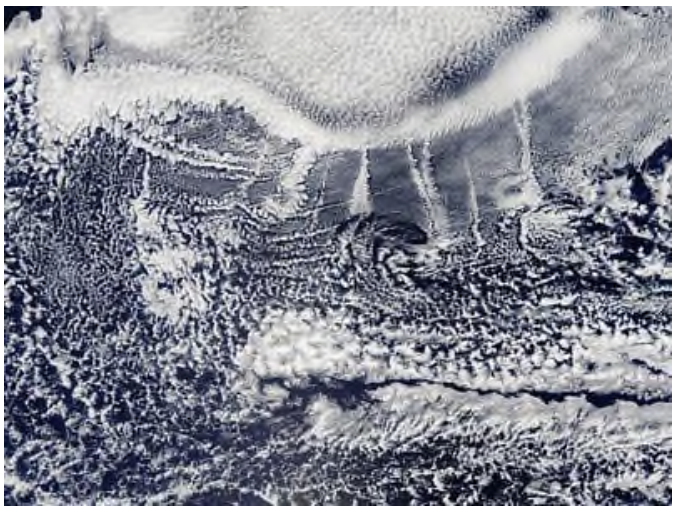

Figure 15. Input image 'cloud_amo_2.0.1.8.2.3.8._lrg.jpg'.

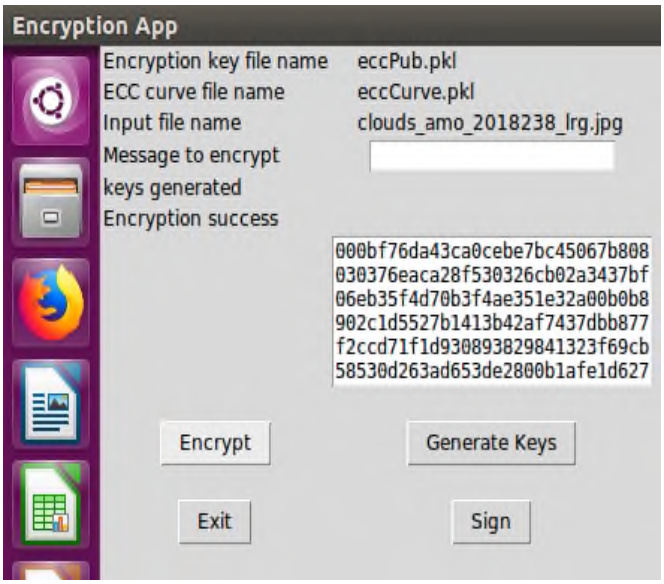

Figure 16. Encryption at the sender.

Figure 17. Signature generation.

At the receiver end, the data is verified by using the private key. The verification procedure is shown in **Figure 18**.

Figure 18. Signature verification and decryption at the receiver.

The image was decrypted using ECC private key pair and retrieved the image, as shown in **Figure 19**.

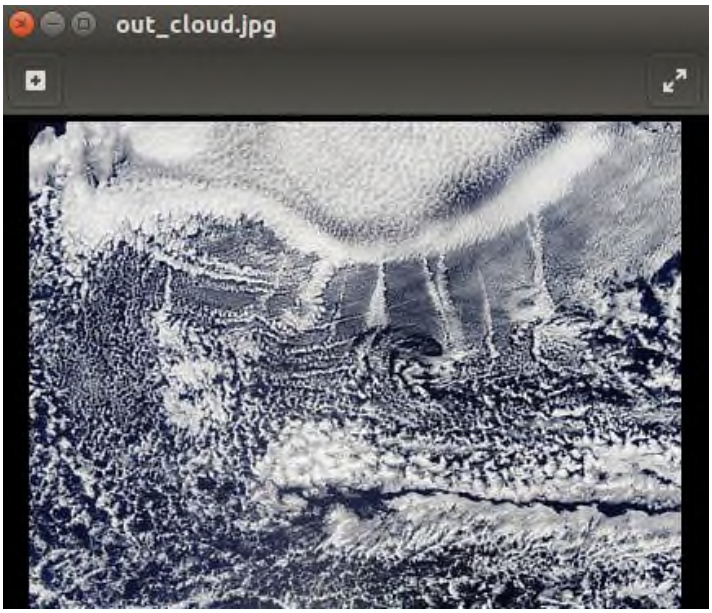

Figure 19. Decrypted image 'out_cloud.jpg'.

Similarly, the space image of campfire data and other stages, such as encrypted by using ECC obtained the hex file, encrypted data signed by ECDSA, the receiver end data verification by using the private key, and the decrypted image using ECC private key pair and image retrieved are also illustrated in **Figure 20**.

The receiver core program accepts the data from the sender and performs encryption, as shown in **Figure 21**. Upon receiving the data through the physical channel, the receiver host transfers it to the data link layer. Within the data link layer, the receiver identifies the service packet and proceeds to check if the spacecraft ID matches. If there is a match, it then verifies the virtual channel ID for a successful match.

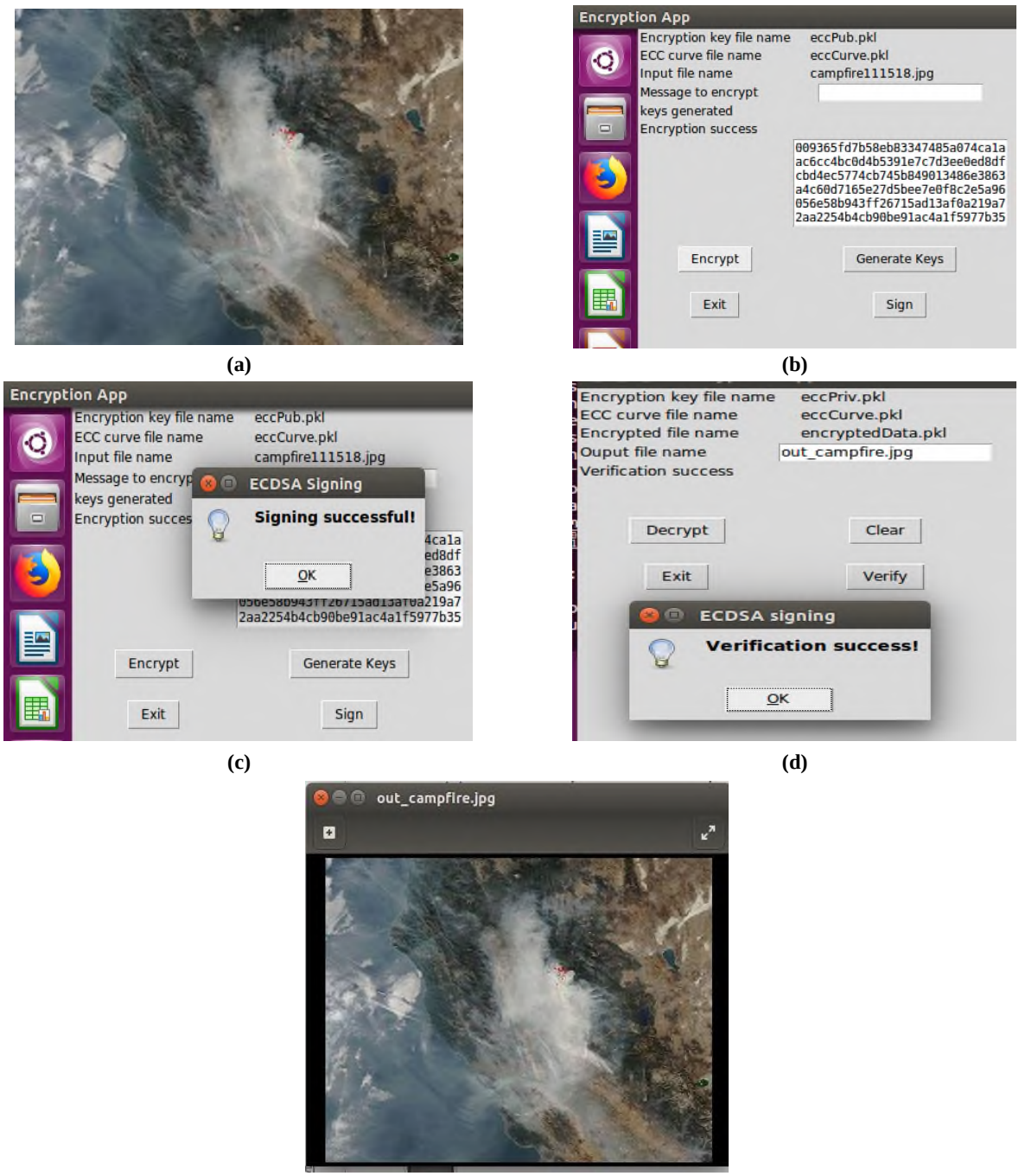

Figure 20. (a) Input image 'campfire111518.jpg'; **(b)** Encryption at the sender; **(c)** Signature generation; **(d)** Signature verification; **(e)** Decrypted image.

Once confirmed, the header is removed, and the data is transmitted to the IP layer. Within the IP layer, the receiver identifies the destination address. Upon identifying the IPv4 header, the IP layer examines the protocol field in the IPv4 header, where a value of 17 indicates UDP. The IP datagram is then passed to the UDP layer. Within the UDP layer, the UDP layer verifies the application port number. If the port number matches, the data is forwarded to the CFDP layer. The CFDP entity receives the data, and the Receiver identifies the metadata details. If the destination entity matches, the receiver utilizes its private key to decrypt the data. Subsequently, the decrypted file data is stored in the file store.

Transmission Mode: Reliable Transmission Mode

Sender

IP Address: 10.0.0.2

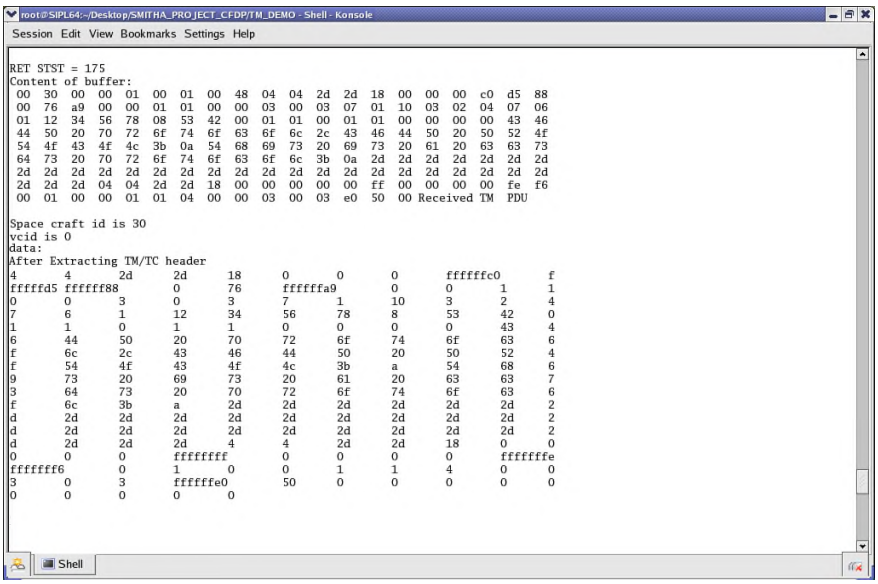

Figure 21. Core procedure at the receiver.

This mode of transmission is reliable and is guaranteed transmission. ACK will be generated after successful transmission. **Figure 22** depicts the user interface by calling put.request service primitive.

The sender-side core mode is shown in **Figure 23**. If NAK is 0, then the checksum value is zero.

Figure 22. User interface program at the sender (reliable).

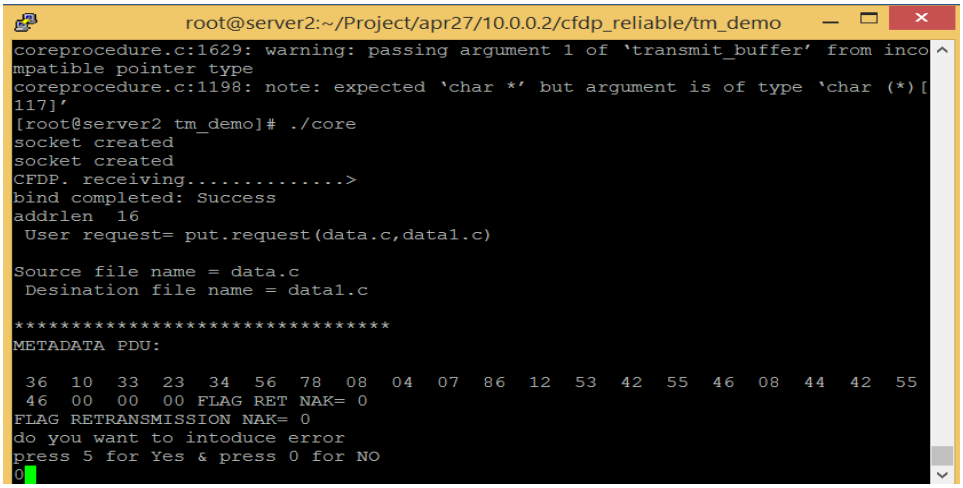

Figure 23. Core procedure output at the sender for reliable transmission.

Core procedure output at the sender for reliable transmission is shown in **Figure 24**. Metadata, file data and CCSDS comprised as CCSDS frame.

Figure 25 shows the CCSDS frame encryption and digital signature generation.

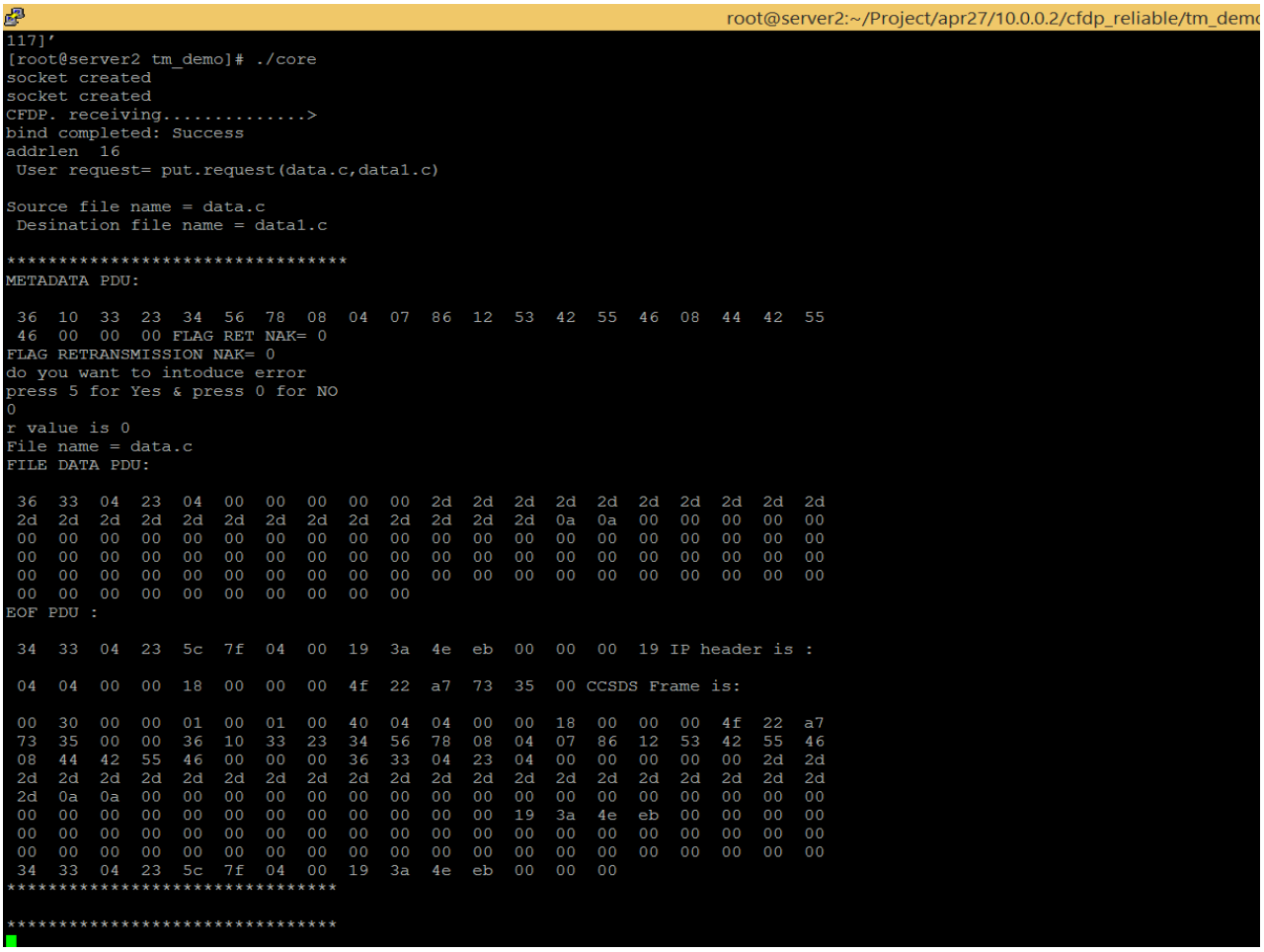

Figure 24. Coreprocedure output at the sender for reliable transmission.

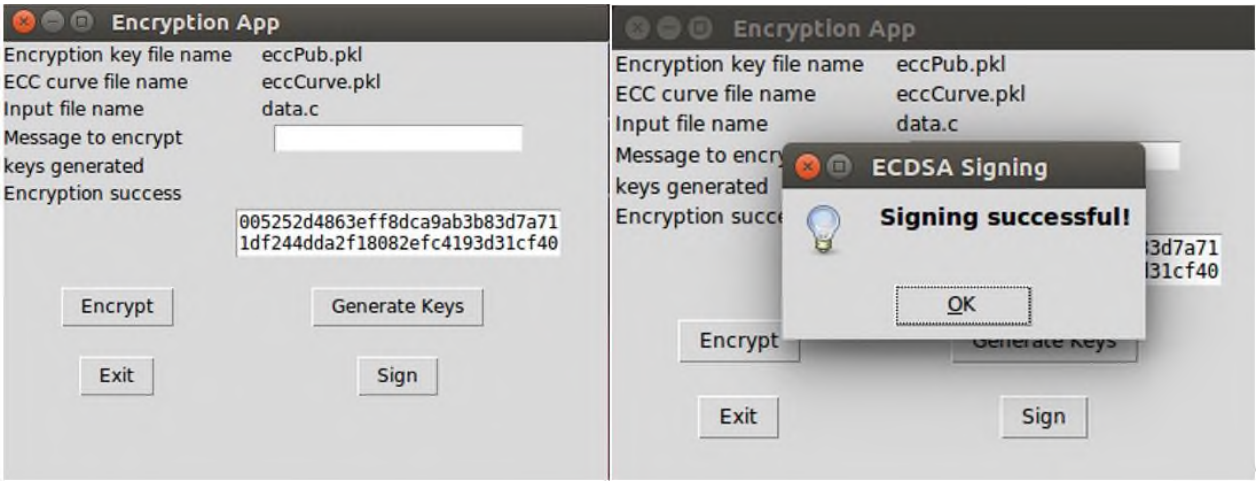

Figure 25. Encryption performed at the sender for reliable transmission.

Receiver

IP Address: 10.0.0.1

At the receiver side, the CCSDS frame is received via space data link and decrypted and verified using ECDSA. Decryption performed at the sender for reliable transmission is shown in **Figure 26**.

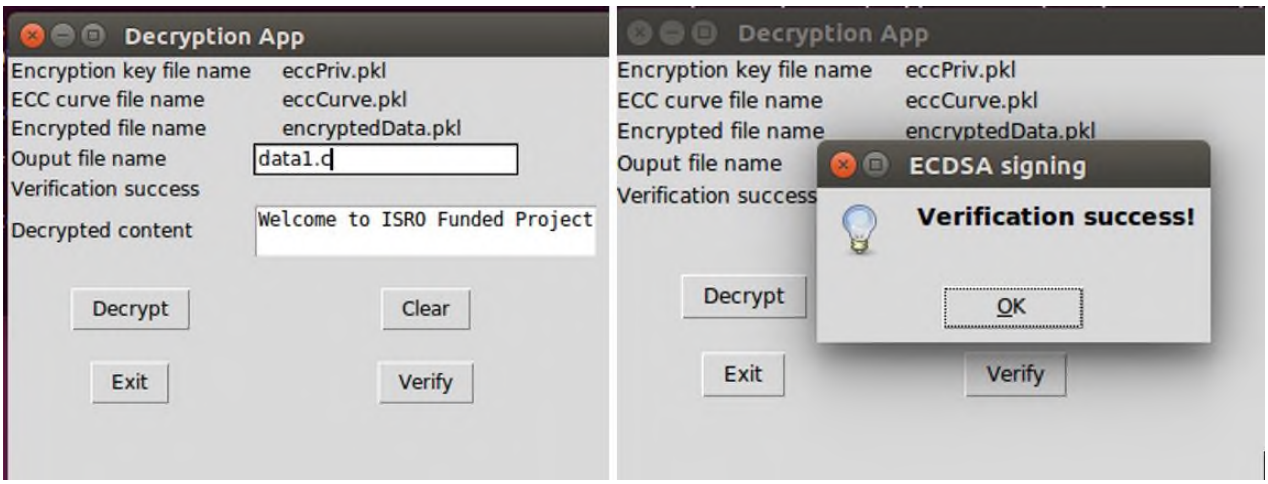

Figure 26. Decryption performed at the sender for reliable transmission.

The receiver core program accepts the data from the sender program after decryption is performed, as shown in **Figure 27**. If the spacecraft id matches, extract the original file data PDU and store it in the file store. The encryption and decryption timing was evaluated by utilizing the provided code, along with a randomly chosen key (k) consisting of 252 integers and a private key (P_k) selected as 325, as demonstrated in **Tables 3** and **4** during the implementation testing. **Table 3** provides encryption and decryption times (in seconds) for different image resolutions. As the image resolution increases, encryption and decryption times also increase. For example, with a resolution of 100×100, encryption takes approximately 6.38 seconds, while decryption takes around 6.17 seconds. However, with higher resolutions, such as 2.56×2.56, encryption time increases to 44.48 seconds, and decryption time rises to 30.91 seconds.

```
root@server1:~/Project/apr27/10.0.0.1/cfdp_reliable_recv/reliabl/TM_DEMO
coreprocedure.c: In function 'form_file_data_pdu':
coreprocedure.c:1255: warning: passing argument 1 of 'fill_linklist' makes integ
er from pointer without a cast
coreprocedure.c:1248: note: expected 'int' but argument is of type 'char *'
[root@server1 TM_DEMO]# ./core
socket created
socket created
socket created
CFDP. receiving.....>
hello
0
connection: Success

Received FILE DATA PDU

Received check sum is 0
19

Received check sum is 19
19
ckecksum ok



 00  30  00  00  01  00  01  00  40  04  04  00  00  18  00  00  00  4f  22  a7
 73  35  00  00  36  10  33  23  34  56  78  08  04  07  86  12  53  42  55  46
 08  44  42  55  46  00  00  00  36  33  04  23  04  00  00  00  00  00  2d  2d
 2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d  2d
 2d  0a  0a  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
 00  00  00  00  00  00  00  00  00  00  00  00  19  3a  4e  eb  00  00  00  00
 00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
 00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
 34  33  04  23  5c  7f  04  00  19  3a  4e  eb  00  00  00 Received TM  PDU
Space craft id is 30
vcid is 0
seq is 2
Both Equal
4       4       0       0       18      0       0       0       4f      22      f
fffffa7 73      35      0       0       36      10      33      23      34      5
6       78      8       4       7       ffffff86        12      53      42      5
5       46      8       44      42      55      46      0       0       0       3
6       33      4       23      4       0       0       0       0       0       2
d       2d      2d      2d      2d      2d      2d      2d      2d      2d      2
d       2d      2d      2d      2d      2d      2d      2d      2d      2d      2
d       2d      2d      a       a       0       0       0       0       0       0
0       0       0       0       0       0       0       0       0       0       0
0       0       0       0       0       0       0       0       0       0       0
0       19      3a      4e      fffffeb         0       0       0       0       0
0       0       0       0       0       0       0       0       0       0       0
0       0       0       0       0       0       0       0       0       0       0
```

Figure 27. (*Continued*).

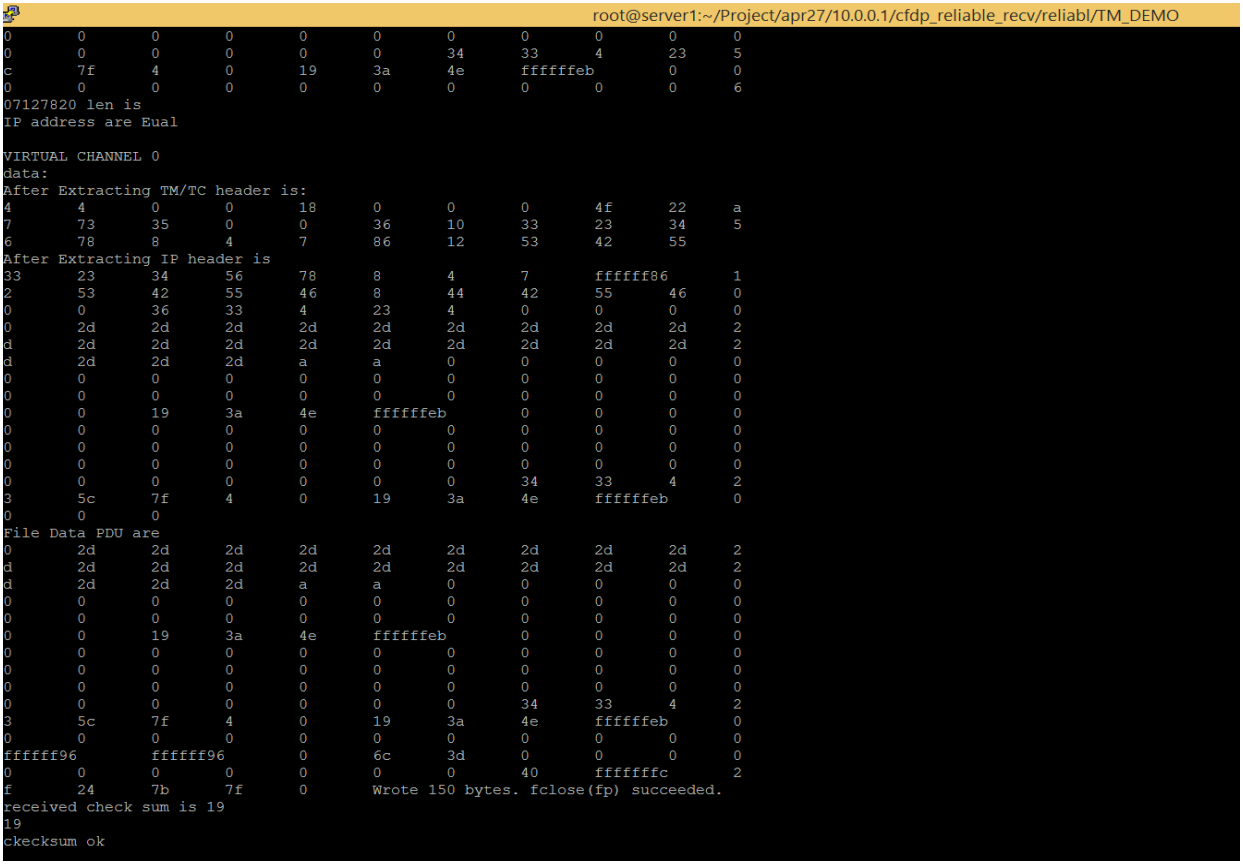

Figure 27. Core procedure output at the receiver for reliable transmission.

Table 3. Examination of the identical security level for a few normally utilized cryptographic key sizes.

Time to break in MIPS years	RSA/DSA key size	ECC key size	RSA/ECC key size ration
10^4	512	106	5:1
10^9	768	132	6:1
10^{11}	1024	160	7:1
10^{20}	2048	210	10:1
10^{79}	21,000	600	35:1

Table 4. Examination of the timing for encryption and decryption.

Image resolution	Encryption time seconds	Decryption time seconds
100×100	6.38	6.17
150×150	13.12	12.52
200×200	43.02	29.88
256×256	44.48	30.91

Table 4 presents the estimated time required to break cryptographic keys in Million Instructions Per Second (MIPS) years, the key sizes for RSA/DSA and ECC, and the ratio between RSA/ECC key sizes. An RSA/DSA, with a key size of 512 bits, would take approximately 104 MIPS years to break. For a key size of 768 bits, the estimated time to break increases to around 109 MIPS years. A key size of 1024 bits would require approximately 1011 MIPS years to break. When using a key size of 2048 bits, the time to break rises to approximately 1020 MIPS years. For a significantly larger RSA/DSA key size of 21000 bits, it would take approximately 1079 MIPS years to break. Comparing the RSA/ECC key sizes, the ratio between RSA and ECC key sizes increases progressively: For a 512-bit RSA/DSA key, the corresponding ECC key size is 106 bits, resulting in a ratio of 5:1. With a 768-bit RSA/DSA key, the ECC key size is 132 bits, resulting in a ratio

of 6:1. For a 1024-bit RSA/DSA key, the ECC key size is 160 bits, resulting in a ratio of 7:1. With a 2048-bit RSA/DSA key, the ECC key size is 210 bits, resulting in a ratio of 10:1. Finally, for a large 21000-bit RSA/DSA key, the ECC key size is 600 bits, resulting in a ratio of 35:1.

8. Conclusions

This research work focuses on ensuring the dependability and security of space file transactions. To enhance security, we have developed an improved elliptic key public key cryptosystem based on curve coordinate system. This cryptographic system enhances the overall security of data transactions. Moreover, the system design also incorporates fault tolerance mechanisms, making the protocol more reliable. We have implemented various fault-handling mechanisms within the CFDP protocol, which further strengthens its reliability. While the Internet Planetary Network (IPN) surpasses CFDP in certain areas, such as network scalability and compatibility with terrestrial delay-tolerant networking technology, it is important to note that the deployed IPN protocols will not replace CFDP but rather complement it. In this project, we analyze the design elements of both technologies and explore ways to integrate them seamlessly. This integration leads to new capabilities, including the efficient transmission of large files through the simultaneous operation of multiple relay satellites.

CFDP serves as a stable global standard that offers significant benefits to mission operations in terms of cost reduction and risk mitigation. By enabling reliable file transfer and remote file system management across interplanetary distances, CFDP plays a crucial role in deep space exploration missions. However, by incorporating emerging delay-tolerant networking technology into interplanetary internet operations, specifically in conjunction with CFDP within complex mission architectures, we can further enhance CFDP's value and effectiveness. This integration holds immense potential for advancing deep space exploration missions. The comparative analysis of Enhanced ECDSA, RSA, and DSA highlights the superiority of ECDSA in delivering digital signatures. The findings reveal the dominance of ECDSA in both the generation of keys and the signing process, where its performance outstripped that of its counterparts. The primary selling point of ECDSA lies in its ability to maintain similar security levels with significantly smaller key sizes. Despite potential vulnerabilities to certain attacks, such as Pollard's Rho and Pohlig-Hellman algorithms, ECDSA remains a more robust choice. The slower processing times of these attacks compared to those against RSA and DSA reinforce the security advantages of ECDSA. The rigorous requirements of ECDSA pertaining to hash functions, discrete logarithms, and number generation bolster its performance claims.

The future expansion of this study should aim to explore the efficacy of ECDSA in real-world applications, particularly focusing on its resistance to emerging threats. It could also further investigate potential enhancements in ECDSA's underlying principles to augment its security performance. Additionally, more exhaustive comparative analyses involving newer cryptographic algorithms may offer additional insight into the evolving dynamics of digital signature technology.

Author contributions

Conceptualization, SS; methodology, SS and SBVS; software, SS and SBVS; validation, SS, SBVS and PM; formal analysis, PM and LA; investigation, PM and LA; data curation, SS, SBVS, PM, and LA; writing—original draft preparation, SS; writing—review and editing, SBV, PM, and LA; visualization, PM and LA; supervision, SS and SBV; project administration, SS and SBVS; funding acquisition, PM and LA. All authors have read and agreed to the published version of the manuscript.

Conflict of interest

The authors declare no conflict of interest.

Abbreviations

CCSDS	Consultative Committee for Space Data Systems
CFDP	CCSDS File Delivery Protocol
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
GF	Galois Field
OBC	On-Board Computer
PDU	Protocol Data Unit
PKC	Public Key Cryptography
MCC	Mission Control Center
NCC	Network Control Centre
TM	Telemetry
TC	Telecommand

References

1. Chauhan SS, Jain N, Pandey SC. Digital signature with message security process. In: Proceedings of the 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE); 28–29 April 2022; Greater Noida, India. pp. 182–187.
2. Genç Y, Afacan E. Design and implementation of an efficient elliptic curve digital signature algorithm (ECDSA). In: Proceedings of the 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS); 21–24 April 2021; Toronto, Canada. pp. 1–6.
3. Mehibel N, Hamadouche M. A new enhancement of elliptic curve digital signature algorithm. *Journal of Discrete Mathematical Sciences and Cryptography* 2020; 23(3): 743–757. doi: 10.1080/09720529.2019.1615673
4. Federal Information Processing Standard publication #180-1. *Secure Hash Standard.* National Institute of Standards and Technology; 1995. p. 310.
5. Federal Information Processing Standards publication #186-2. *Digital Signature Standard (DSS)*. National Institute of Standards and Technology; 2000. pp. 309–312.
6. Recommendations Regarding Federal Information Processing Standard (FIPS) #186-2. *Digital Signature Standard (DSS)*. National Institute of Standards and Technology; 2001. p. 311.
7. Chen L, Moody D, Regenscheid A, Randall K. *Recommendations for Discrete Logarithm-Based Cryptography: Elliptic Curve Domain Parameters.* National Institute of Standards and Technology; 2019.
8. Brickell E, Pointcheval D, Vaudenay S, Yung M. Design validations for discrete logarithmbased signature schemes. In: Imai H, Zheng Y (editors). *Public Key Cryptography. PKC 2000. Lecture Notes in Computer Science.* Springer; 2000. Volume 1751. pp. 276–292.
9. Brown DRL. The exact security of ECDSA. Available online: https://cacr.uwaterloo.ca/techreports/2000/corr2000-54.ps (accessed on 30 August 2023).
10. Kim SR, Kyung R. Study on modified public key cryptosystem based on elgamal and cramer-shoup cryptosystems. In: Proceedings of the 2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC); 2023; 8–11 March 2023; Las Vegas, NV, USA. pp. 0280–0284.
11. Zhou R, Lin Z. An improved exponential elgamal encryption scheme with additive homomorphism. In: Proceedings of the 2022 International Conference on Blockchain Technology and Information Security (ICBCTIS); 15–17 July 2022; Huaihua, China. pp. 25–27.
12. ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley GR, Chaum D (editors). *Advances in Cryptology. CRYPTO 1984. Lecture Notes in Computer Science*. Springer; 2000. Volume 196. pp. 469–472.
13. Koblitz N. CM-Curves with good cryptographic properties. In: Feigenbaum J (editor). *Advances in Cryptology—CRYPTO'91. CRYPTO 1991. Lecture Notes in Computer Science*. Springer; 2001. Volume 576. pp. 279–287.
14. Silverman JH. An introduction to thetheory of elliptic curves. Available online: https://www.math.brown.edu/~jhs/Presentations/WyomingEllipticCurve.pdf (accessed on 21 July 2023).
15. Huang J, Jiao J, Wang Y, et al. Age-critical long erasure coding-CCSDS file delivery protocol for dual-hop S-IoT. *IEEE Internet of Things Journal* 2023. doi: 10.1109/JIOT.2023.3274519

16. Clarke NL, Katos V, Menesidou SA et al. A Novel Security Architecture for a Space-Data DTN. In: Koucheryavy Y, Mamatas L, Matta I, et al. (editors). *Wired/Wireless Internet Communication. WWIC 2012*. *Lecture Notes in Computer Science*, Springer; 2012. Volume 7277. pp. 31–45.
17. Liu Z, Liu R, Zhang H, et al. Parallel implementation of the CCSDS turbo decoder on GPU. In: *China Communications (Early Access)*. IEEE; 2023. pp. 1–8.
18. Gupta K, Silakari S. Performance analysis for image encryption using ECC. In: Proceedings of the 2010 International Conference on Computational Intelligence and Communication Networks; 26–28 November 2010; Bhopal, India. pp. 79–82.
19. Aswatha AR, Sasi S, Santhosh B, et al. Design and implementation of unreliable CFDP protocol over elliptic curve cryptography. In: Satapathy S, Bhateja V, Mohanty J, et al. (editors). *Smart Intelligent Computing and Applications, Smart Innovation, Systems and Technologies*. Springer; 2019. Volume 160. pp. 627–638.
20. Khan NA, Awang A. Elliptic curve cryptography for the security of insecure internet of things. In: Proceedings of the 2022 International Conference on Future Trends in Smart Communities (ICFTSC); 1–2 December 2022; Kuching, Sarawak, Malaysia. pp. 59–64.
21. Fang X, Wu Y. Investigation into the elliptic curve cryptography. In: Proceedings of the 2017 3rd International Conference on Information Management (ICIM); 21–23 April 2017; Chengdu, China. pp. 412–415.
22. Ali Z, Ghani A, Khan I, et al. A robust authentication and access control protocol for securing wireless healthcare sensor networks. *Journal of Information Security and Applications* 2020; 52: 102502. doi: 10.1016/j.jisa.2020.102502
23. Pushpa SX, Raja KS. Enhanced ECC based authentication protocol in wireless sensor network with DoS mitigation. *Cybernetics and Systems* 2022; 53(8): 734–755. doi: 10.1080/01969722.2022.2055403
24. Srividya BV, Akhila S. Implementing a hybrid crypto-coding algorithm for an image on FPGA. In: Satapathy S, Joshi A (editors). *Information and Communication Technology for Intelligent Systems (ICTIS 2017)-Volume 2*. Springer; 2017. pp. 72–84.
25. Mahto D, Yadav DK. RSA and ECC: A comparative analysis. *International Journal of Applied Engineering Research* 2017; 12(19): 9053–9061.