Received: 13 October 2023 | *Revised:* 21 March 2024 | *Accepted:* 19 July 2024
DOI: 10.37917/ijeee.22.1.30 *Vol. 22 | Issue 1 | June 2026*

Iraqi Journal for Electrical and Electronic Engineering
Original Article

Open Access
IJEEE University of Basrah College of Engineering

Performance Analysis of Elliptic Curve Cryptography Digital Signature Algorithm

Mohammed Wameedh Abdulnabi*[1], Raad A. Muhajjar[1], Mishall Al-Zubaidie[2]
[1]Department of Computer Science, College of Computer Science and Information Technology, University of Basrah, Basrah, Iraq
[2]Department of Computer Science, College of Education for Pure Science, University of Thi-Qar, Nasiriyah 64001, Iraq

Correspondance
*Mohammed Wameedh Abdulnabi
Department of Computer Science, College of Computer Science and Information Technology,
University of Basra, Basra, Iraq
Email: itpg.mohammed.wameedh@uobasrah.edu.iq

Abstract
Because elliptic curve cryptography offers a promising trade-off between security and computational performance, the field of current cryptographic techniques has taken a particular interest in it. Two basic digital signature algorithms—the Elliptic Curve Diffie-Hellman Algorithm (ECDH) and the Elliptic Curve Digital Signature Algorithm (ECDSA) are the subject of this performance analysis and comparison. These methods are based on elliptic curve cryptography. The analysis takes into consideration realistic application demands as well as factors like key length and security level. The results provide useful information on trade-offs between performance and security. A list of acceptable ECDSA requirements for digital signatures was used for the comparison. These characteristics are sign, sign/s, no PC verify, no PC verify/s, siglen, keygen, keygen/s, verify, and verify/s.

Keywords
Cryptography, DH, ECC, ECDH, ECDSA.

I. Introduction

Electronic signatures, often referred to as digital signatures or electronic signatures, are used to confirm that documents and transactions made online are valid. Electronic signatures are widely used to verify the user's identity and the validity of the transaction in a range of applications, including online banking services. There are several unique characteristics of an electronic signature. A digital signature is an individual's property. It is not possible to assign or transfer a right [1].

The use of electronic signatures in electronic financial systems is not without its problems, though, particularly when it comes to security and efficiency. Conventional electronic signature systems need complex hardware and encryption techniques, which can be time- and resource-consuming. Consequently, using it might not be possible to use electronic banking apps, particularly on mobile devices or in environments with constrained bandwidth [2]. The scientists recommended making use of to solve these problems, lightweight electronic signature systems were developed specifically for use in electronic banking applications. Usually, these systems are based on simple algorithms that function well on many devices, such as tablets and smartphones. A simple electronic signature method is the Digital Signature Algorithm (DSA), which is based on hash functions and standard arithmetic [3]. Among the various applications where DSA is commonly used are electronic financial services. It is thought to be safe and efficient [4]. A crucial component of cryptocurrency security is the encryption mechanism, which falls into three categories: hashing, symmetric, and asymmetric. The public and secret keys are generated from prime numbers. The primary methods for verifying data are asymmetric encryption techniques and hash functions and protecting cryptocurrency transactions [5]. The maximum key size for symmetric encryption methods like Rivest-Shamir-Adleman (RSA) and Advanced Encryption Standard (AES) is 32 bits, and they are limited in their capacity to generate single keys. Because of

this, there's a chance that attackers may make duplicate copies of these keys, which might compromise prime factors [6].

The alignment of the public and private keys in a digital signature technique may be created to guarantee a safe transaction [7]. Digital signatures also make use of a few algorithms. ECDSA provides an additional example based on elliptic curve cryptography [8]. ECDSA is well known for its effectiveness and security and is frequently utilized in electronic banking and other applications [9] and [10]. On ECC, a great deal of study has been conducted. To provide the same degree of security as RSA or key exchange algorithms like Diffie Hellman, ECC uses fewer keys. ECC has received a lot of attention and acceptance following twenty years of study and development. Laws pertaining to the government, finance, and business have been implemented to support this efficientThe use of public-key technologies increased. In contrast to methods often referred to as symmetric or secret key techniques, like DES, Diffie and Hellman initially proposed the notion of developing new cryptographic systems in 1976. They accomplished this by finding solutions to difficult mathematical problems (such genuine factor problems or discrete logarithms over a finite range). After 10 years, Miller and Kobletz discovered that elliptic curves might present complex problems and possibly usher in a new age of public-key encryption systems. The Elliptic Curve Cipher (ECC) cannot provide the same level of security as RSA or Diffie-Hellman because of the consistency problem. Another reason to discourage cryptanalysis from cryptography is provided by ECC, the next-generation public key cryptography technology [11]. Second, without requiring any pre-existing secrets, ECDH serves as an essential agreement mechanism that makes it possible for two parties to create a shared secret via an insecure communication channel. It functions through the employment of the Diffie-Hellman key exchange protocol and finite-field arithmetic [12] and [13]. In this study, these two algorithms are compared.

II. Related Works

In this article, a novel mapping technique for elliptic curve cryptography (ECC) data encoding and decoding is proposed. This approach uses a lookup table based on the coordinate values' frequency distribution to turn text messages or audio samples into points on an elliptic curve. For real-time applications, the suggested solution is more effective, safe, and appropriate [14]. In several research, the popular digital signature algorithm ECDSA has been assessed and contrasted with alternative signature algorithms. In [15], the researchers offered a comprehensive examination of the ECDSA method's applications, security, and performance. The reason this algorithm was selected is because it works well in many different applications, including electronic banking, electronic health care, electronic government, and electronic commerce, while also providing security. The ECDSA method for electronic applications provides three important security features: integrity, authentication, and non-repudiation. ECDSA has also been shown to work well since it uses short keys and requires less computing power than other public key encryption algorithms like RSA. This research offers a novel, secure, and confidential block chain-based biomedical.

Image processing system in [16]. The final Healthcare 4.0-enabled multimedia imaging processing system consists of the block chain layer, edge layer, fog computing layer, and cloud storage layer. Periodically, the edge layer gathers and transmits patient medical data from the bottom layer to the top layer. Fog nodes securely store multimedia data from the edge layer in cloud storage by using a block chain and lightweight cryptography. Users in the medical field can then securely search for such data to monitor or treat medical issues. For safeguarding biomedical image processing while maintaining privacy, elliptic curve digital signatures (ECDSA), elliptic curve cryptography (ECC), and elliptic curve Diffie-Hellman (ECDH) are suggested. Publicly available CT scans and chest X-rays are used to validate the suggested procedure. The results of the experiments indicate that the proposed model performs better computationally in terms of mean square error (MSE), encryption and decryption durations, and peak-to-signal noise ratio (PSNR). Using various key sizes, Md. Ismail Jabiullah et al. [17] created and developed an ECDSA-based security method for block chain transactions. The researchers provide a novel approach for quick programming implementation of an elliptic curve digital signature technique for self-affirmation, using an estimated starting coefficient p in a finite Galois field GF(p). The most important feature of this approach is that it moves from bit-level operations, which are slow on chips, to word-level operations, which are substantially quicker. In order to complete many higher levels, the mathematical operations carried out during execution replace word-level tasks with bit-level ones. Schematic findings supporting the assertion that ECDSA is suitable under forced situations are obtained with an 8.2 GHz Pentium 4 CPU. A more dependable technique for a wireless sensor network system is employed in this study by [18]. Elliptic Curve Diffie Hellman key exchange mechanism is studied and explored using Elliptic Curve Integrated Encryption Scheme and PyCryptodome module. The advantages of ECC for key creation and encryption and decryption are assessed by contrasting it with Rivest-Shamir-Adleman (RSA) in this study. The results of the investigation demonstrate that ECC has far smaller keys than RSA and provides a greater level of security, allowing for more compact implementations for a certain level of protection. The authenticity of the data becomes increasingly important in data communication systems when

messages are exchanged across unsafe routes. If there is a vulnerability in the transmission system, there is a chance that an untrustworthy party will intercept. Since the elliptic curve that GF (p) depicts is only closed to addition, the consequence of adding two points on an elliptic curve is always another point on the elliptic curve. The authors of [19] utilized p = 149 for their investigation. Higher security is offered by elliptic curve encryption (ECC), which uses a lower key size than non-ECC encryption. For example, a 160-bit ECC key is equally secure as a 1024-bit RSA key. The use of elliptic curve cryptography (ECC) for protecting multi-factor systems in cloud computing. The paper presents a method for ECC encryption and decryption and compares the performance and efficiency of the ECC algorithm with that of the RSA algorithm. This research provides an architecture and flowchart for the proposed system and assesses the secrecy, integrity, and authentication performance of ECC. The findings demonstrate that when it comes to cloud data security, the ECC technique outperforms RSA [20]. This essay provides several email security options, addresses a variety of email communication issues, and describes how email works. It offers several ideas and techniques for enhancing and preserving email system security [21].

III. Proposed Method

Maintaining ECDSA's security and efficacy is now one of our top priorities. It is challenging in many ways to ensure usability across several domains. The purpose of the suggested system is to compare and assess the elliptic curve cipher digital signature algorithm's usability and user experience, including its documentation, user interface, and simplicity, as well as to identify any methods to improve it. The following is the way the research is conducted: The procedure for contrasting data results between the employed algorithms, ECDSA and ECDH, as the sender transmits diverse data types including messages, videos, or other forms of data, is illustrated in the provided Fig. 1.

Fig .2 summarizes the process of digital signatures using asymmetric cryptosystems. Before generating a signature for a message, the sender must first employ cryptography to generate a hash value for it. This value is subsequently encrypted using the sender's private key to produce the signature. The technique below can be used by a third party, such as the receiver, who has access to the signer's details and the message, to validate the signature. The recipient is responsible for decrypting the message using the sender's public key from the signature.

The receiver is also in charge of creating a hash value, h', from the message. The signature is deemed to be genuine if h=h' [22].

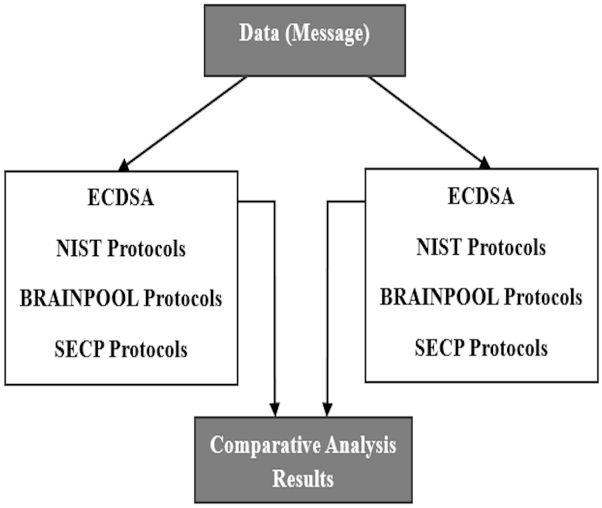


Fig. 1. Proposed Method

A. ECDSA

Elliptic curves are employed in the ECDSA, a variation of the Digital Signature Algorithm (DSA). Using the ECC method, ECDSA generates and verifies digital signatures [23].

B. ECDH

An anonymous key exchange protocol called Elliptic Curve Diffie-Hellmann Key Exchange (ECDH) gives communication users a pair of keys—the public key and the private key—to encrypt data transferred across an insecure channel [24]. The performance of the ECDH and ECDSA may be compared in the following areas.

1) NIST Protocols

The National Institute of Standards and Technology is known by its acronym, NIST. The organization is in responsible of creating and upholding standards for a wide range of technologies, including cryptography. The aforementioned duties are among its responsibilities. Several elliptic curve standards, most notably NIST192p, have been released by NIST. The purpose of these guidelines is to guarantee the effectiveness and safety of elliptic curves [25]. The details that are included in the header of Table I include the following: Length of signature, Key Generation Time: The amount of time required to generate a pair of keys. Keys Generated per Second: The number of keys that are capable of being generated in a single second. signatures on data, and so forth. The number of signatures executed in a second is measured in signatures per second. checking signatures, and other things. Verifications per Second: The quantity of signatures that can be verified in a single second on a personal computer without the need for pre-calculating keys. Signature without a key Verifications

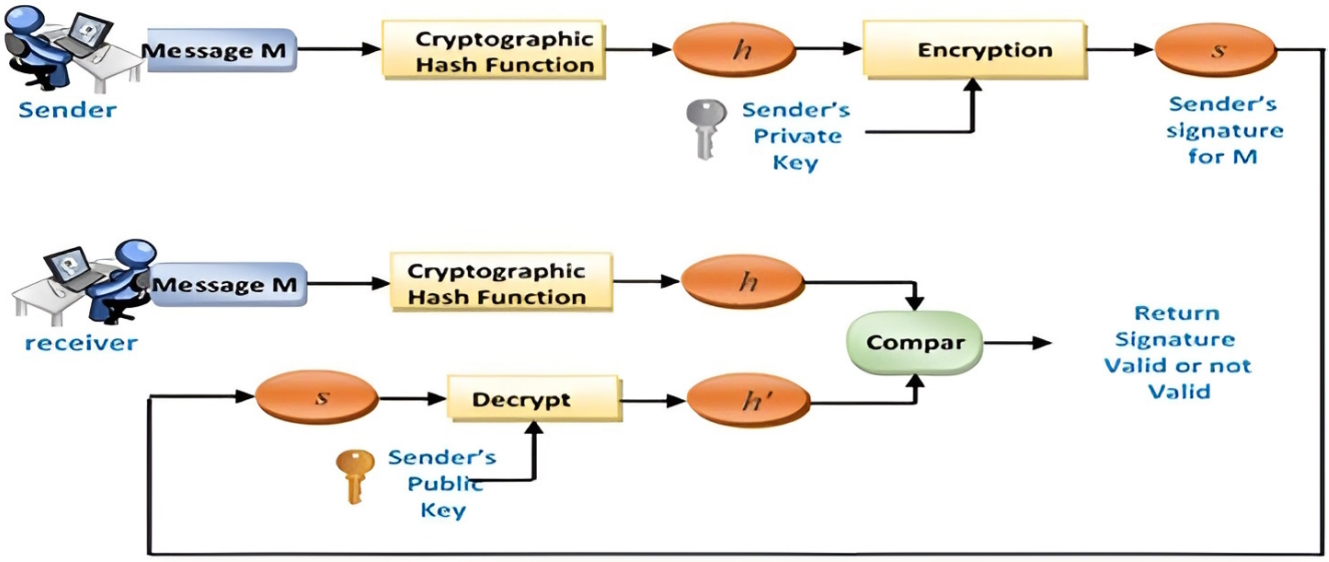


Fig. 2. Essential Elements of Digital Signature Process [19]

per Second: The speed at which a signature may be verified in a single second without the need for a key. The shared secret key and ECDH/s—the frequency of shared secret key derivations—are calculated. ECDH stands for Elliptic Curve Diffie-Hellman per second. This tabledescribes the ECDSA and ECDH performance benchmarks that are based on several NIST hashing protocol characteristics. These are the descriptions of the NIST protocols:

- NIST192p
  NIST192p is a popular elliptic curve in ECDSA because of its small size and good computational performance. It's also crucial to remember that this curve is considered secure, demonstrating how difficult it is to compromise the ECDSA method and determine the secret key from the public key. The NIST192p curve used in ECDSA's equation 1 is as follows [26]:

$$y^2 = x^3 - 3x + 41 \tag{1}$$

  This is an example of an elliptic curve equation, which is a mathematical formula used to generate curve points. There is a finite set of possible points since the NIST192p curve operates inside a finite field that is defined by a prime number of elements. Fig. 3(a) shows the results of the NIST192p curve's implementation. Following are some more specifics regarding the NIST192p curve:

  - The curve consists of 2192, 264, 232, and 977 prime elements.
  - The curve has an infinite point.
  - The curve has a generator point, or a point from which further points on the curve can be generated.
  - The cofactor, or the number that indicates how many points are on the curve, is present.

The NIST192p curve is an example of an elliptic curve on a 192-bit field. The (2) given below for Elliptic Curve Diffie-Hellman (ECDH) operations controls its properties:

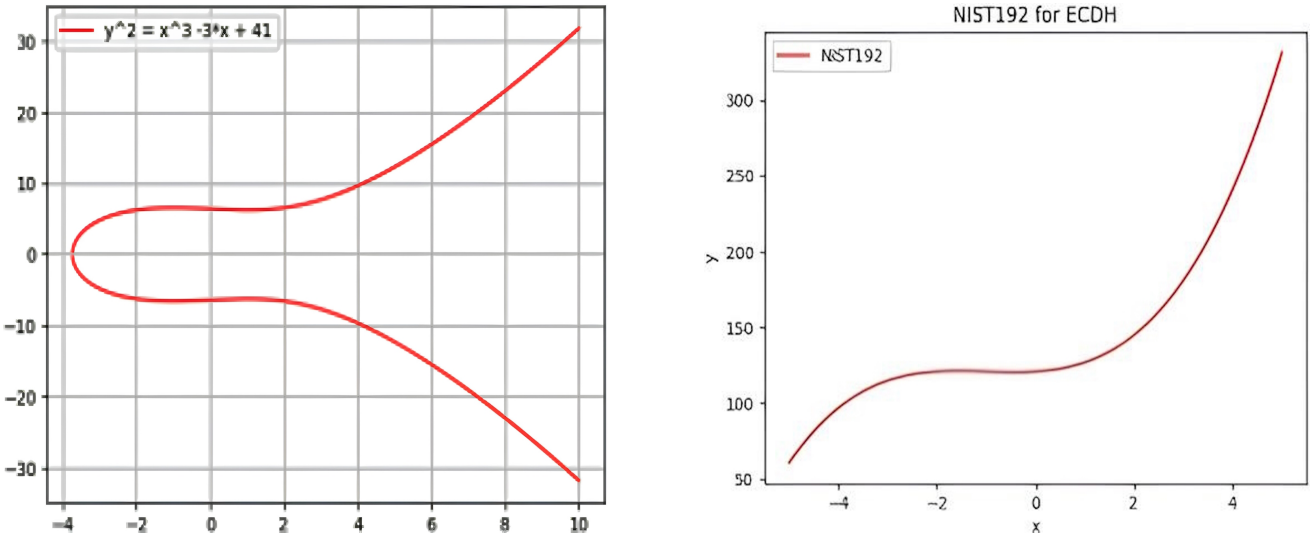
$$y^2 = x^3 + 486662x^2 + x + 17 \tag{2}$$

In ECDH, a key exchange protocol that generates shared keys using elliptic curves, NIST192p, is utilized. As a reliable and effective key exchange mechanism, ECDH is gaining popularity. The NIST192p curve's implemented findings are shown in Fig. 3(b). Here is an illustration of how keys can be exchanged using ECDH:

1. Alice and Bob have chosen to utilize the NIST192p curve in accordance with their agreement.
2. Bob and Alice produce a public key that corresponds to a private key.
3. Alice and Bob alternately exchange public keys.
4. Alice uses Bob's public key to generate a shared key.

TABLE I. ECDSA AND ECDH ALGORITHMS RESULTS FOR NIST PROTOCOLS

Method	Siglen	Keygen	Keygen/s	Sign	Sign/s	Verify	Verify/s	No PC Verify	No PC Verify/s	Ecdh	Ecdh/s
NIST192p	48	0.00059s	1698.20	0.00062s	1618.15	0.00105s	948.58	0.00190s	526.78	0.00141s	706.83
NIST224p	56	0.00056s	1777.08	0.00060s	1674.26	0.00104s	958.04	0.00233s	429.84	0.00171s	583.47
NIST256p	64	0.00065s	1550.36	0.00065s	1530.41	0.00134s	749.02	0.00259s	386.02	0.00198s	504.48
NIST384p	96	0.00103s	974.22	0.00108s	926.29	0.00246s	406.95	0.00477s	209.79	0.00338s	295.54
NIST521p	132	0.00191s	524.76	0.00183s	547.92	0.00344s	291.00	0.00702s	142.54	0.00547s	196.98



(a) NIST192p ECDSA Curve

(b) NIST192p ECDH Curve

Fig. 3. NIST192p ECDSA and ECDH curves [19]

5. Bob uses Alice's public key to produce a key that both of them agree on.

6. Alice and Bob succeeded in generating a shared key that enabled them to both encrypt and decode data.

ECDH stands as a secure and efficient method for exchanging keys, and its popularity is on the rise. The remaining NIST protocols function in a similar manner.

- NIST224p [27]

  The NIST224p curve's equation (3), which is utilized in ECDSA to produce points on a curve:

$$y^2 = x^3 - 3x + 89 \tag{3}$$

  The following equation (4) of the NIST224p curve used in ECDH to generate points on a curve:

$$y^2 = x^3 - 3x^2 + 4x + 6 \tag{4}$$

- NIST256p

  NIST256p Points on a curve can be created using the NIST256p curve equation (5) used in ECDSA [28] [29]:

$$y^2 = x^3 - 3x + b \tag{5}$$

  Here, b is a constant that equals: 41943038089353123056564681522243970785283756427907490438260516314151816149433 6. The NIST256p curve equation (6) used in ECDH can be used to create points on a curve [30]:

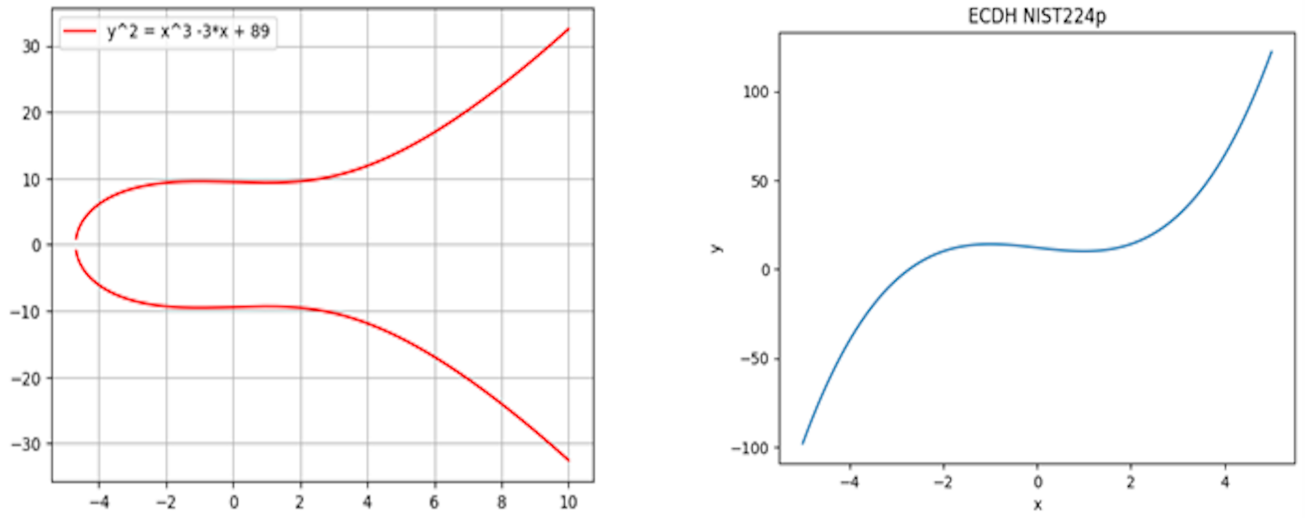
$$y^2 = x^3 - 3^2 + 4x + 6 \tag{6}$$

  Figures 4 and 5 ((a) and (b)) show the implemented NIST224p/NIST256p curve results for the ECDSA algorithm and the ECDH algorithm, accordingly.

- NIST384p

  The NIST384p curve's equation (7), which is used in the ECDSA, is as follows [19]:

$$y^2 = x^3 - 3x + b \tag{7}$$



(a) NIST224p ECDSA Curve

(b) NIST224p ECDH Curve

Fig. 4. NIST224p ECDSA and ECDH curves

The value of b is: 48271253523239869562012854564739278971123449533398611214232422252252224234 5492, where b is a constant. In ECDH, the NIST384p curve's equation (8) is as follows:

$$y^2 = x^3 - 3x^2 + 4x + 6 + 2^3 84 - 1 \quad (8)$$

Fig. 6 (a) displays the results of the implemented NIST384p curve for the ECDSA algorithm, while (b) displays the results of the implemented NIST384p curve for the ECDH algorithm.

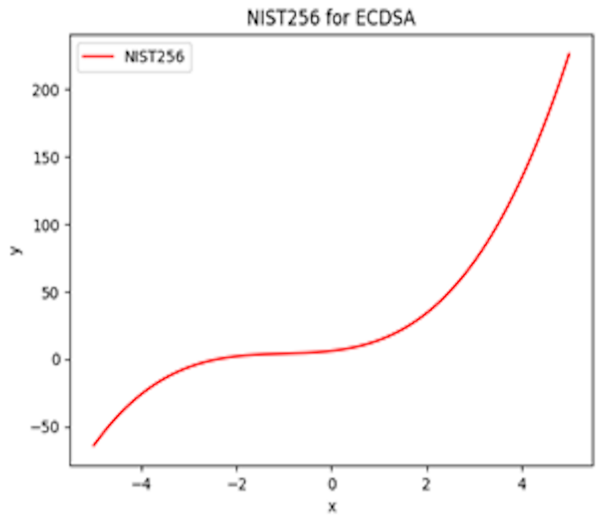
- NIST521p The NIST521p curve equation (9) used in ECDSA is as follows [19]:

$$y^2 = x^3 - 3x^2 + 4x + 6 + 2^5 21 - 1 \quad (9)$$

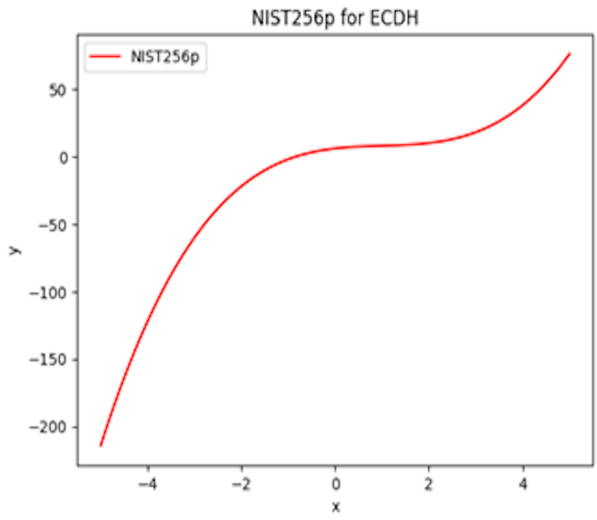
Fig. 7 (a) displays the results of the implemented NIST521p curve for the ECDSA method, while (b) displays the results of the implemented NIST521p curve for the ECDH algorithm.

Fig. 8, which compares ECDSA with ECDH utilizing several of the previously discussed properties.

1. Siglen, the length of a signature:
   - **ECDSA**: The chosen curve affects how long the signature is. The signature length for NIST192p, for instance, is 48 bytes.
   - **ECDH**: Since ECDH doesn't create signatures, there is no such thing as a signature length.
2. Key Generation (keygen):
   - The ECDSA key creation process involves the development of a private key and a corresponding public key.
   - As part of the ECDH key creation process, a private key and matching public key are formed.
3. Speed of Key Generation (keygen/s):
   - **ECDSA**: The key generation speed fluctuates according to the selected curve. For example, NIST192p has a key production rate of about 1700 keys per second.
   - **ECDH**: The selected curve determines how quickly keys are created. For example, NIST192p has a key production rate of about 1700 keys per second.
4. Signing (sign):
   - **ECDSA**: Signing comprises utilizing the private key to create a signature for a particular message.
   - **ECDH**: This method does not require a signature.
5. Signing Speed (sign/s):
   - **ECDSA**: Depending on the curve being utilized, the signing pace varies. For instance, NIST192p can sign documents at a rate of about 1600 signatures per second.



(a) NIST 256p ECDSA Curve



(b) NIST 256p ECDH Curve

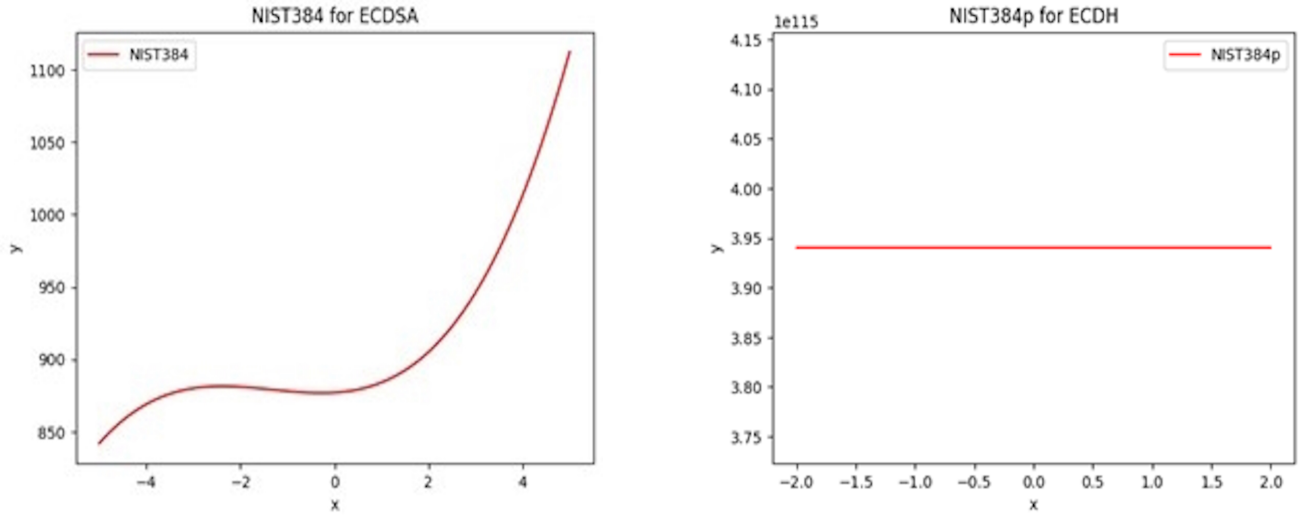
Fig. 5. NIST256p ECDSA and ECDH curves

  - **ECDH**: There is no need for a signature using this procedure.
6. Verification (verify):
  - **ECDSA**: As part of the verification procedure, ECDSA employs the signature and public key to verify the validity of a communication.
  - **ECDH**: There is no verification in ECDH.
7. Verification Speed (verify/s):
  - **ECDSA**: Depending on the curve being used, the verification speed varies. The verification speed for NIST192p is around 900 verifications per second.
  - **ECDH**: There is no verification in ECDH.
8. No PC verification:
  - **ECDSA**: Provides verification without the need for a personal computer (PC).The used curve shows how quickly a PC can be verified. For example, NIST192p verifies at a rate of about 560 per second.
  - **ECDH**: ECDH does not require PC-based verification.Elliptic Curve Diffie-Hellman, or ECDH, is a technique that allows two parties to share a secret key via an unreliable channel. ECDH is not used for signing or verifying
9. No PC Verification Speed (no PC verify/s):
  - **ECDSA**: The used curve shows how quickly a PC can be verified. For example, NIST192p verifies at a rate of about 560 per second.
  - **ECDH**: ECDH does not require PC-based verification.
10. ECDH (Elliptic Curve Diffie-Hellman):
  - **ECDH**: A technique that allows two parties to share a secret key via an unreliable channel. ECDH is not used for signing or verifying.
11. ECDH Speed (ecdh/s):
  - The curve being used affects how quickly ECDH operations take place. For instance, NIST192p has an ECDH speed of about 700 key exchanges per second.

2) BRAINPOOL Protocols

The Brainpool Standardization Working Group formulated the brainpool curves as an alternative to the NIST curves, and these curves have now been standardized. Brainpool P160r1 can be used for cryptographic operations such as key exchange (ECDH) and digital signatures (ECDSA). Its goal is to strike a compromise between computing efficiency and security. Additional protocols include Brainpoolp192r1, Brainpoolp224r1, Brainpoolp256r1, Brainpoolp320r1, Brainpoolp384r1, Brainpoolp512r1, and Brainpoolp192r1 (192 bits), among others [31]. Table II compares the performance of ECDH and ECDSA. BRAINPOOLP160r1: The Brainpoolp160r1 curve's equation (5) is what is utilized in ECDSA [23]: The value of b is: 1566083949407573529996955224135760342422259061068512044368000095328. The Brainpoolp160r1 curve used in ECDH has the following equation (10):

$$y^2 = x^3 + x + 7 \tag{10}$$



(a) NIST384p ECDSA Curve

(b) NIST384p ECDH Curve

Fig. 6. NIST384p ECDSA and ECDH curves

Fig.9 (a) shows the Brainpoolp160r1 curve's implementation results for the ECDSA technique, and (b) shows the Brainpoolp160r1 curve's implementation results for the ECDH methodology.

- BRAINPOOLP192r1
  The Brainpoolp192r1 curve's equation (5) for the ECDSA [23] where b is a constant that has the following values: 4827125352323986956201285456473927897112344953339861121423242225225222423454 93 Fig.10 (a) displays the results of the implemented Brainpoolp192r1 curve for the ECDSA method, while (b) Displays the results of the implemented Brainpoolp192r1 curve for the ECDH algorithm.

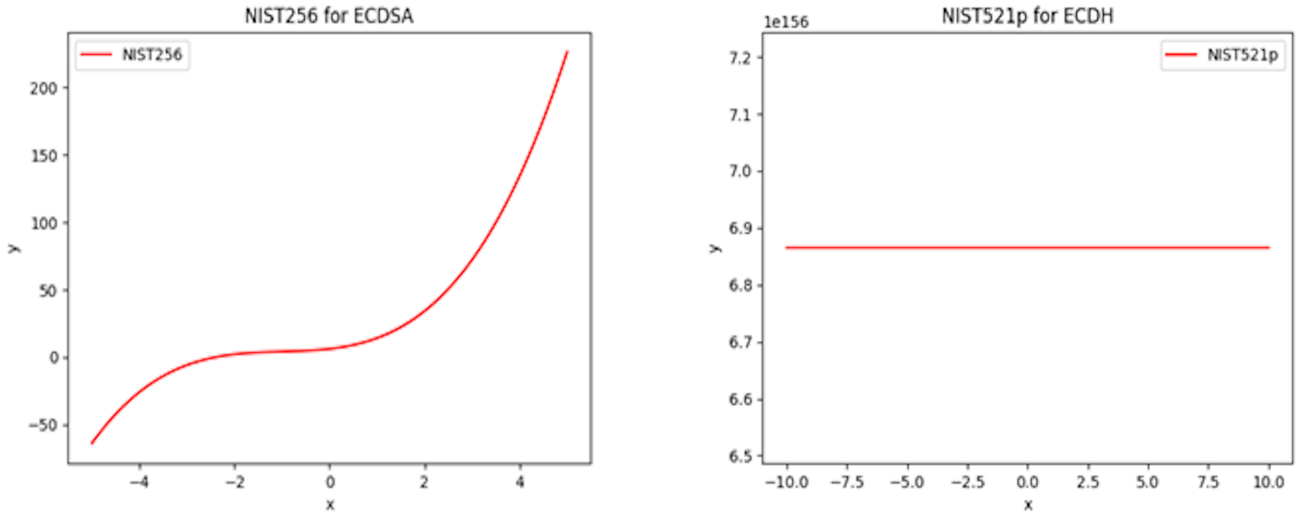
- BRAINPOOLP224r1
  The Brainpoolp224r1 curve's equation (5) for the ECDSA is [23] where b is a constant with the value of: 265443576 9789423065727343008115773299969552241357603424 2225903600113.
  Fig.11 (a) displays the results of the implemented Brainpoolp224r1 curve for the ECDSA method, while (b) displays the results of the implemented Brainpoolp224r1 curve for the ECDH algorithm.

- BRAINPOOLP256r1
  The Brainpoolp256r1 curve's equation (5) for ECDSA is [23] Parameter b is a constant equal to: 419430380893 5312305654648152224397078528375642790749043826051631415181614943 36. Fig.12(a) illustrates the outcomes of the implemented Brainpoolp256r1 curve concerning the ECDSA algorithm, whereas Fig.12(b) showcases the outcomes of the implemented Brainpoolp256r1 curve concerning ECDH algorithm.

- BRAINPOOLP320r1
  In ECDSA, the Brainpoolp320r1 curve's Equation is (5) where b is a constant that has the value 2320 - 2192 - 296 + 264 - 1 in this situation [23]. Fig.13 (a) displays the results of the implemented brainpoolp320r1 curve for the ECDSA method, while (b) displays the results of the implemented Brainpoolp320r1 curve for the ECDH algorithm

- BRAINPOOLP384r1
  The equation (5) of the Brainpoolp384r1 curve used in ECDSA is [23]: b is a constant that has the value 2384 - 2256 + 2160 - 296 + 232 - 1 in this situation . Fig.14 (a) displays the results of the implemented Brainpoolp384r1 curve for the ECDSA method, while (b) Displays the results of the implemented Brainpoolp384r1 curve for the ECDH algorithm.

- BRAINPOOLP512r1
  The equation (5) of the Brainpoolp512r1 curve used in ECDSA is [23]: The Brainpoolp512r1 curve's ECDSA [23]Equation (5) is: b is a constant with the value of 2512 - 2256 + 2160 - 296 + 232 - 977 in this case. Fig. 15 (a) displays the results of the implemented Brainpoolp512r1 curve for the ECDSA method, while (b) Displays the results of the implemented Brainpoolp512r1 curve for the ECDH algorithm.



(a) NIST521p ECDSA Curve

(b) NIST521p ECDH Curve

Fig. 7. NIST521p ECDSA and ECDH curves

TABLE II. ECDSA AND ECDHA ALGORITHMS RESULTS FOR BRAINPOOL PROTOCOLS

Method	Siglen	Keygen	Keygen/s	Sign	Sign/s	Verify	Verify/s	No PC Verify	No PC Verify/s	Ecdh	Ecdh/s
BRAINPOOLP160r1	40	0.00038s	2609.41	0.00041s	2428.87	0.00081s	1229.20	0.00160s	625.24	0.00133s	749.6 9
BRAINPOOLP192r1	48	0.00046s	2172.62	0.00049s	2024.56	0.00100s	928.33	0.00209s	478.66	0.00165s	606.5 4
BRAINPOOLP224r1	56	0.00069s	1459.70	0.00068s	1504.76	0.00120s	829.80	0.00236s	424.07	0.00222s	451.0 0
BRAINPOOLP256r1	64	0.00063s	1573.71	0.00067s	1591.17	0.00129s	773.17	0.00256s	390.36	0.00232s	430.9 4
BRAINPOOLP320r1	80	0.00082s	1218.40	0.00087s	1142.10	0.00170s	588.90	0.00333s	300.29	0.00265s	376.6 5
BRAINPOOLP384r1	96	0.00105s	955.16	0.00119s	838.11	0.00293s	341.60	0.00469s	213.42	0.00379s	263.9 6
BRAINPOOLP512r1	128	0.00200s	481.66	0.00182s	548.33	0.00324s	308.86	0.00736s	135.85	0.00564s	177.4 5

From Fig. 16, which compares ECDSA with ECDH utilizing many of the features previously presented.

Brainpool protocols' efficiency in ECDSA and ECDH functions might differ according on which particular elliptic curve is used. This overview provides a brief synopsis of the performance comparison based on the used attributes.

- ECDSA verification usually takes longer than ECDSA signature procedure (sign).
- Signature pace (sign/s): Depending on the curve being used, the signing pace varies. For instance, ECDSA-brainpoolP256r1 can sign approximately 1600 signatures per second.
- Verification procedure (verify): Compared to the ECDSA signature procedure, the ECDSA verification operation is typically slower.
- Verification Speed (verify/s): Depending on the chosen curve, the verification speed varies. For instance, ECDSA-brainpoolP256r1 verifies at a rate of about 750 times per second.
- ECDSA provides verification without a personal computer (PC), which is a feature.
- Speed of Verification without a PC (no PC verify/s): Depending on the curve applied, verification without a PC can be performed. quickly. For instance, ECDSA-brainpoolP256r1 verifies transactions at a rate of about 300 per second.

The ECDH's performance can be summarized as follows:

- Key Generation Speed (keygen/s): The curve being employed affects the key generation speed. For instance, ECDHE-secp256r1 may generate keys at a rate of about 1600 key exchanges per second.
- ECDH Speed (ecdh/s): The chosen curve affects how quickly ECDH operations proceed. For instance, ECDHE-secp256r1 has an average ECDH speed of 400 key exchanges per second.

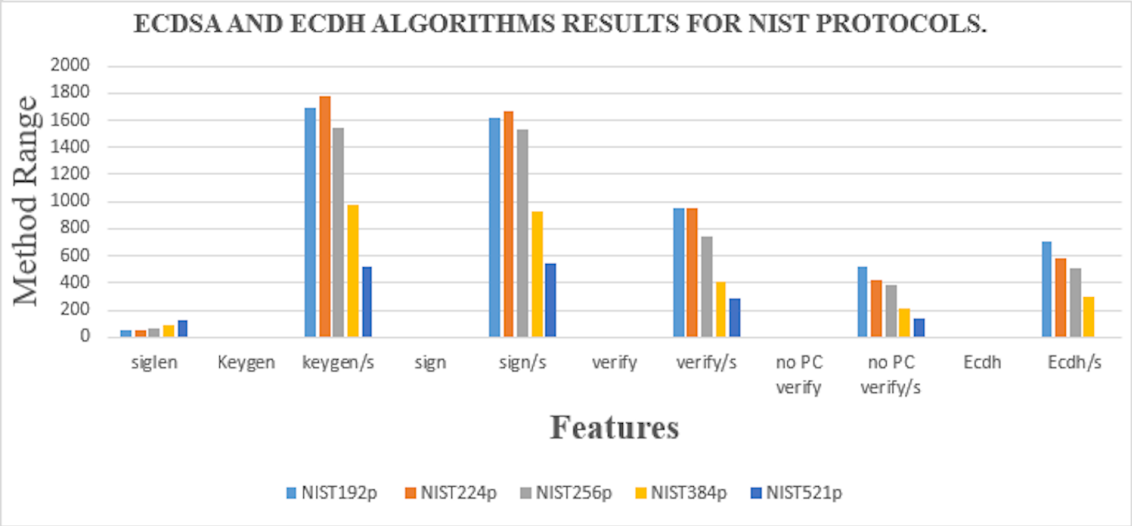


Fig. 8. NIST Protocols comparison for ECDSA and ECDH features.

- Depending on the particular elliptic curve employed, the performance of PRAINPOOL protocols in ECDSA and ECDH operations may differ. The performance comparison based on the employed characteristics is summarized as follows:
- ECDSA Generally speaking, signing operations (sign) take less time than ECDSA verification..
- Signature pace (sign/s): Depending on the curve being used, the signing pace varies. For instance, ECDSA-brainpoolP256r1 can sign approximately 1600 signatures per second.
- Verification procedure (verify): Compared to the ECDSA signature procedure, the ECDSA verification operation is typically slower.
- Verification Speed (verify/s): Depending on the chosen curve, the verification speed varies. For instance, ECDSA-brainpoolP256r1 verifies at a rate of about 750 times per second.
- ECDSA provides verification without a personal computer (PC), which is a feature.
- Speed of Verification without a PC (no PC verify/s): Depending on the curve applied, verification without a PC can be performed quickly. For instance, ECDSA-brainpoolP256r1 verifies transactions at a rate of about 300 per second.

It is important to recognize that a number of variables, including the specific implementation, hardware platform, and optimization techniques used, can affect how effective Brainpool protocols are. Furthermore, the performance comparison may differ for various elliptic curves and protocol implementations.

3) SECP Protocols

The Standards for Efficient Cryptography Group (SECG) defines the SECP curves, which stand for the Standards for Efficient Cryptography. SECP comes in a number of variants, including SECP112r1, SECP112r2, SECP128r1, SECP160r1, and SECP256k1.Comparability between the six curves is high. The main difference is the size of the field [32]. Table III compares the effectiveness of ECDH and ECDSA.

- SECP112r1
  This is the SECP112r1 curve's equation (5) [24]. Where the value of the constant b is $2^{112} - 2^{96} + 2^{32} - 977$. The SECP112r1 curve for ECDH's equation (11) is as follows:

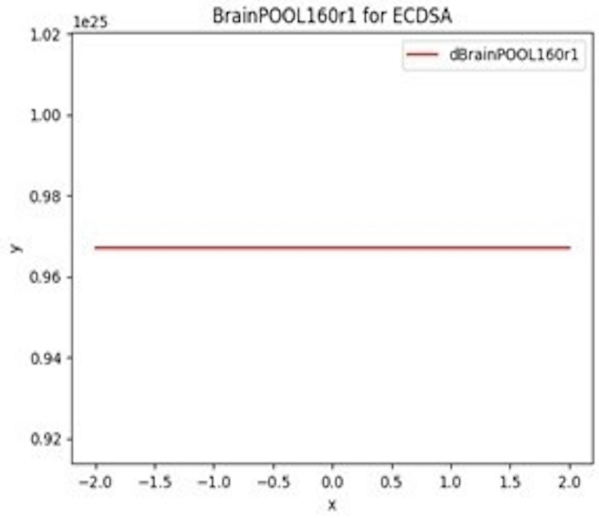
$$y^2 = x^3 + 2x^2 + 1 \tag{11}$$

  Fig. (17a) displays the SECP112r1 curve implementation results for the ECDSA method, whereas (b) displays the SECP112r1 curve implementation results for the ECDH algorithm.

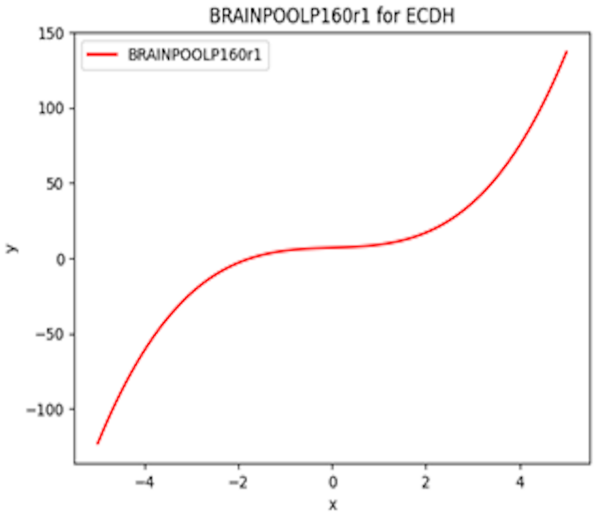
- SECP112r2
  SECP112r2's elliptic curve, equation (12) [24]:

$$y^2 = x^3 - 3x + 861 \tag{12}$$

  Fig. 18 (a) and (b) demonstrate the results of the implementation of the SECP112r2 curve for the ECDSA and ECDH algorithms, respectively.



(a) Brainpoolp160r1 ECDSA Curve



(b) Brainpoolp160r1 ECDH Curve

Fig. 9. Brainpoolp160r1 ECDSA and ECDH curves

TABLE III. RESULTS OF THE ECDSA AND ECDHA ALGORITHMS FOR SECP PROTOCOLS

Method	Siglen	Keygen	Keygen/s	Sign	Sign/s	Verify	Verify/s	No PC Verify	No PC Verify/s	Ecdh	Ecdh/s
SECP112r1	28	0.00027s	3683.58	0.00029s	3437.16	0.00056s	1794.78	0.00100s	1004.39	0.00077s	1300.46
SECP112r2	28	0.00025s	4021.27	0.00028s	3591.01	0.00058s	1714.98	0.00093s	1075.69	0.00077s	1295.75
SECP128r1	32	0.00031s	3269.48	0.00032s	3145.78	0.00056s	1796.05	0.00109s	918.64	0.00091s	1094.53
SECP160r1	42	0.00040s	2487.10	0.00042s	2357.96	0.00077s	1303.17	0.00153s	655.15	0.00121s	827.69
SECP256k1	64	0.00061s	1648.48	0.00067s	1482.59	0.00124s	809.19	0.00245s	408.41	0.00211s	0.002 11s

- SECP128r1
  The SECP128r1 elliptic curve Equation (13) is [24]:

$$y^2 + x^3 + 486662X^2 + X - 1 \tag{13}$$

  Fig 19 (a) displays the SECP128r1 curve implementation results for the ECDSA method, whereas (b) displays the SECP128r1 curve implementation results for the ECDH algorithm.

- SECP160r1
  The SECP160r1 curve's ECDSA Equation (14) is [24]:

$$y^2 = x^3 + 7 \tag{14}$$

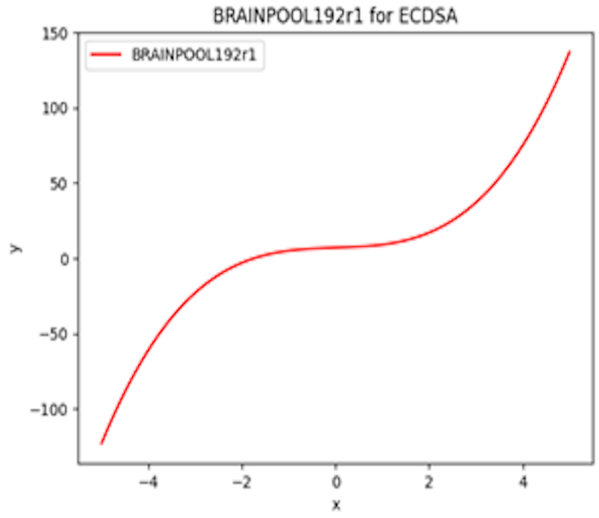
  Fig. 20 (a) displays the SECP160r1 curve implementation results for the ECDSA method, whereas Fig. 20 (b) displays the SECP160r1 curve implementation results for the ECDH algorithm.

- Secp256k1
  ECDSA curves employ equation (14) of secp256k1 as follows [24]. Fig.21 SECP256K1 shows the implemented results of curve for ECDSA algorithmin (a) and shows the implemented results of SECP256K1 curve for ECDH algorithmin (b).

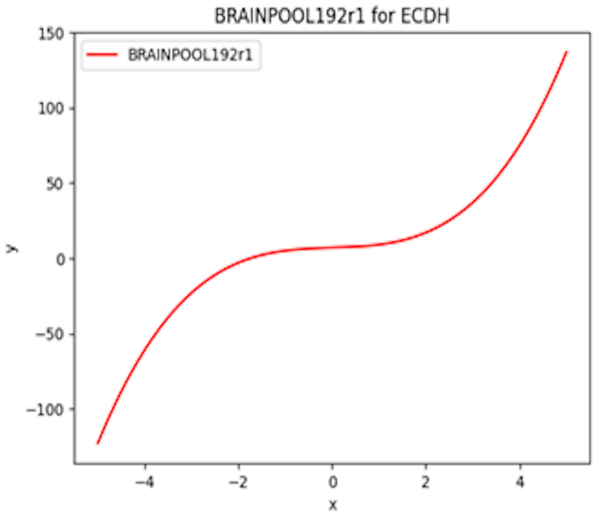
From Fig. 22, which compares ECDSA with ECDH utilizing many of the features previously presented.

Fig.22 shows the following results:.

1. **Signature Length (siglen):**
   - ECDSA's signature length is determined by the curve it uses. The signature length for SECP160, for instance, is 42 bytes.
   - Since ECDH doesn't create signatures, there is no such thing as a signature length.
2. **Key Generation (keygen):**
   - ECDSA key generation entails the production of a matching public key as well as a private key.
   - A private key and the related public key must be created as part of the ECDH key generation process.



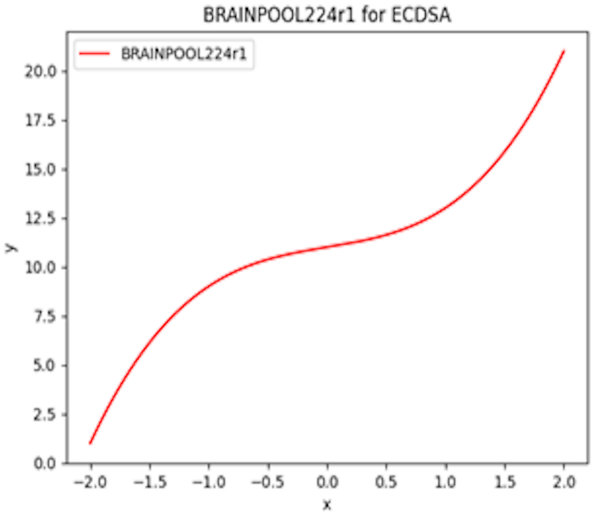
(a) Brainpoolp192r1 ECDSA Curve



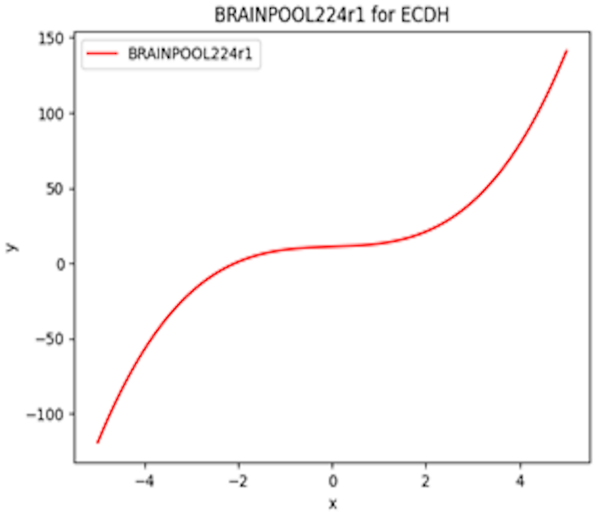
(b) Brainpoolp192r1 ECDH Curve

Fig. 10. Brainpoolp192r1 ECDSA and ECDH curves

3. **Key Generation Speed (keygen/s):**
   - Depending on the chosen curve, the key generation speed varies. For instance, SECP160 can generate keys at a rate of about 2500 keys per second.
   - ECDH: The selected curve determines how quickly keys are created. For example, ECDHE-secp256r1 has a key generation rate of about 1600 key exchanges per second.
4. **Signing (sign):**
   - ECDSA: Signing includes developing a signature for a specific message using the private key.
   - ECDH: This method does not require a signature.
5. **Signing Speed (sign/s):**
   - The curve being used affects the signature speed. For instance, the signing rate for ECDSA-SECP256 is approximately 1400 signatures per second.
   - ECDH: This method does not require a signature.
6. **Verification (verify):**
   - ECDSA: Verification entails examining a message's authenticity using the public key and the signature.
   - ECDH: There is no verification in ECDH.
7. **Verification Speed (verify/s):**
   - ECDSA: Depending on the chosen curve, the verification speed varies. For instance, the verification rate for ECDSA-SECP256 is approximately 800 verifications per second.
   - ECDH: There is no verification in ECDH.
8. **No PC Verification:**
   - ECDSA: Verification is possible using this approach without requiring a personal computer (PC).
   - ECDH: ECDH does not require PC-based verification.
9. **No PC Verification Speed (no PC verify/s):**
   - ECDSA: The speed of verification without a computer is impacted by the selected curve. ECDSA-SECP256, for example, verifies at a rate of about 400 times per second.
   - ECDH: ECDH does not require PC-based verification.
10. **ECDH (Elliptic Curve Diffie-Hellman):**
    - The selected curve determines how quickly the ECDH process continues. Using an unprotected channel, two parties can generate a shared secret key by using ECDH for key exchange. For example, ECDHE-secp160r1 has an ECDH performance of about 700 key exchanges per second.

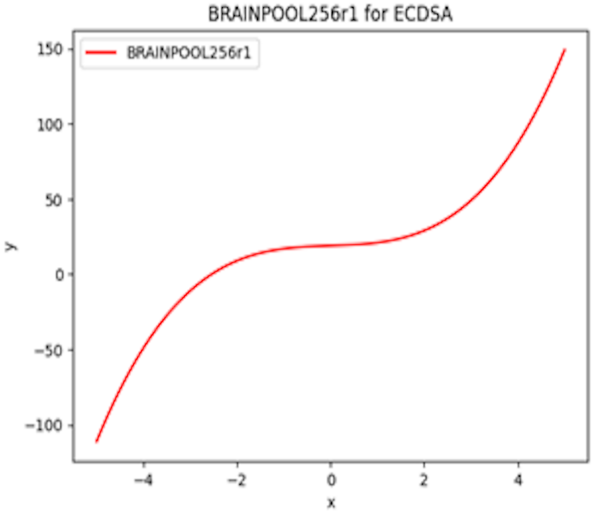


(a) Brainpoolp224r1 ECDSA Curve

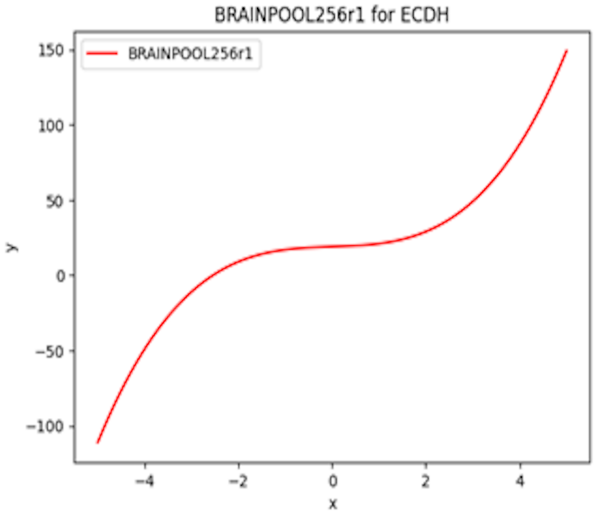


(b) Brainpoolp224r1 ECDH Curve

Fig. 11. Brainpoolp224r1 ECDSA and ECDH curves

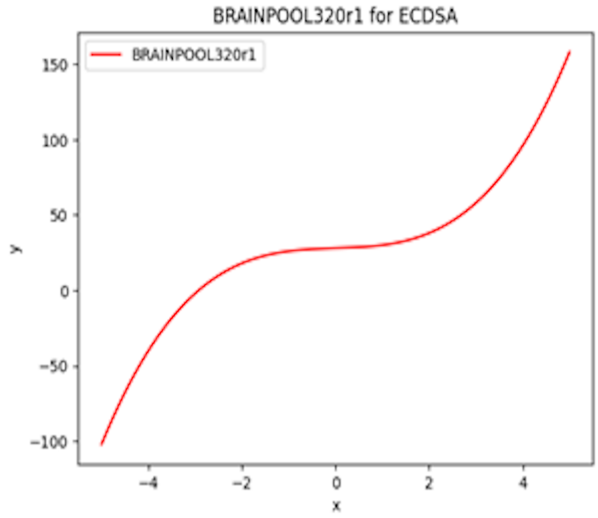


(a) Brainpoolp256r1 ECDSA Curve

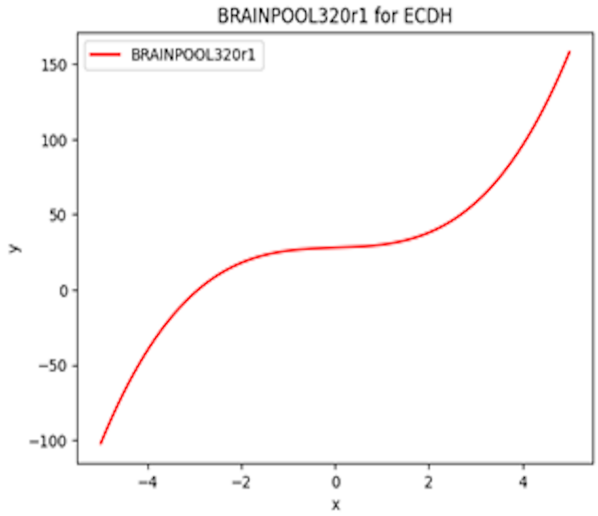


(b) Brainpoolp256r1 ECDH Curve

Fig. 12. Brainpoolp256r1 ECDSA and ECDH curves

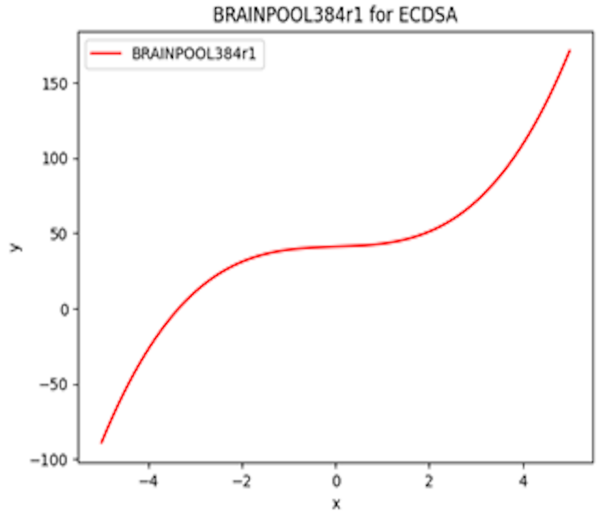


(a) Brainpoolp320r1 ECDSA Curve

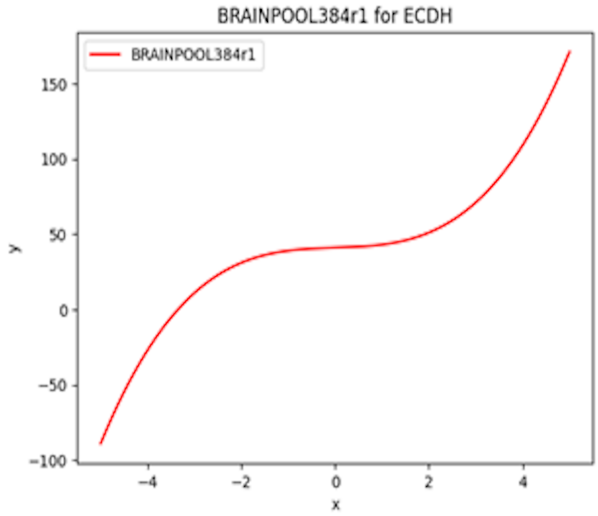


(b) Brainpoolp320r1 ECDH Curve

Fig. 13. Brainpoolp320r1 ECDSA and ECDH curves

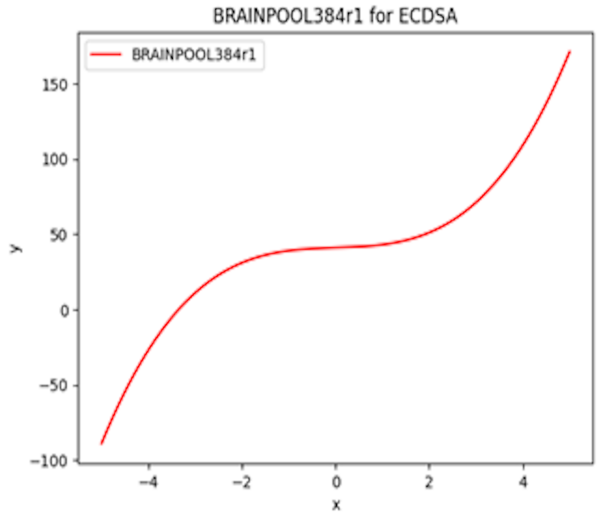


(a) Brainpoolp384r1 ECDSA Curve

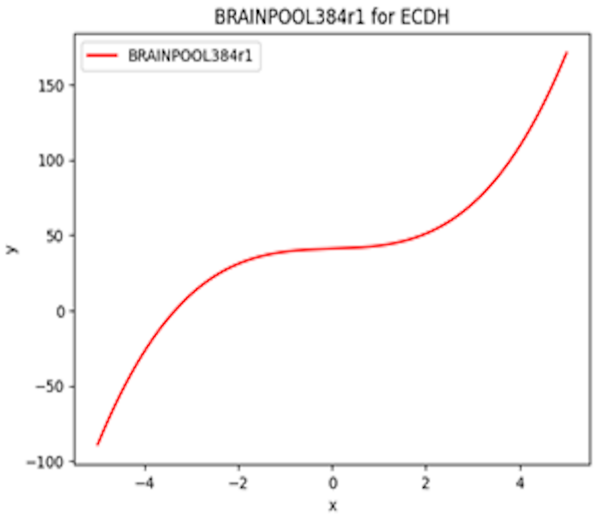


(b) Brainpoolp384r1 ECDH Curve

Fig. 14. Brainpoolp384r1 ECDSA and ECDH curves



(a) Brainpoolp512r1 ECDSA Curve



(b) Brainpoolp512r1 ECDH Curve

Fig. 15. Brainpoolp512r1 ECDSA and ECDH curves

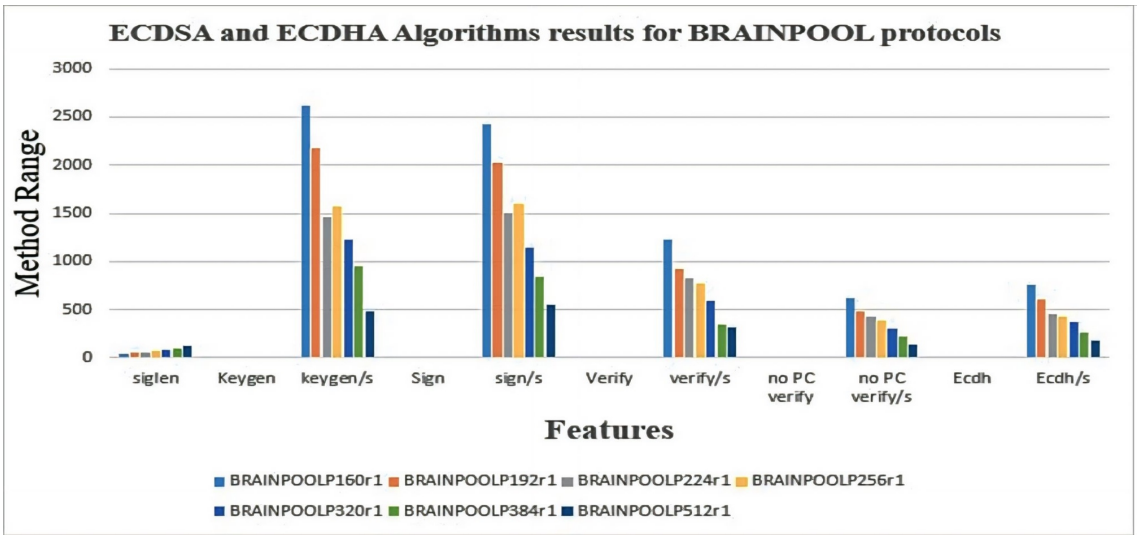
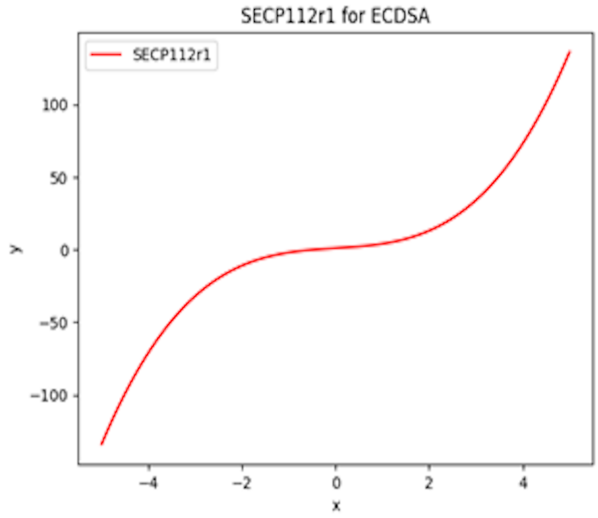
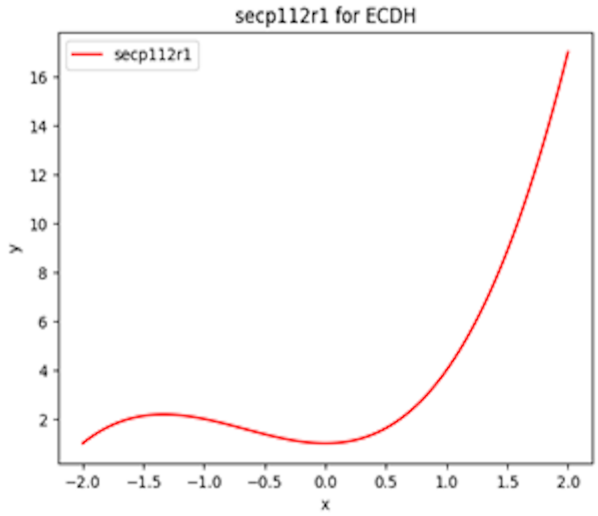


Fig. 16. Comparing the Brainpool Protocols for ECDSA and ECDH Features.

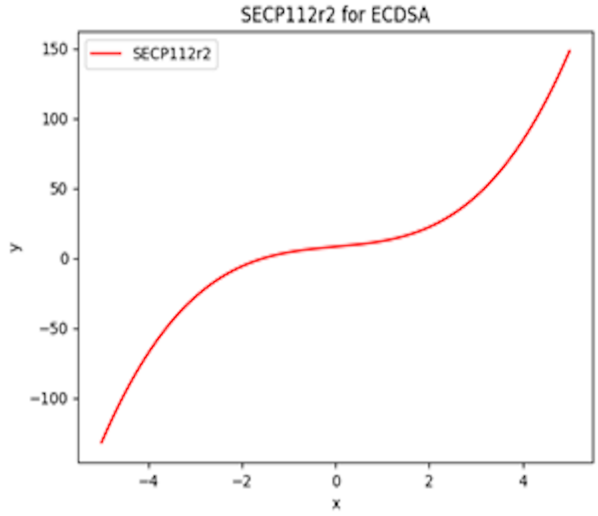


(a) SECP112r1A ECDSA Curve

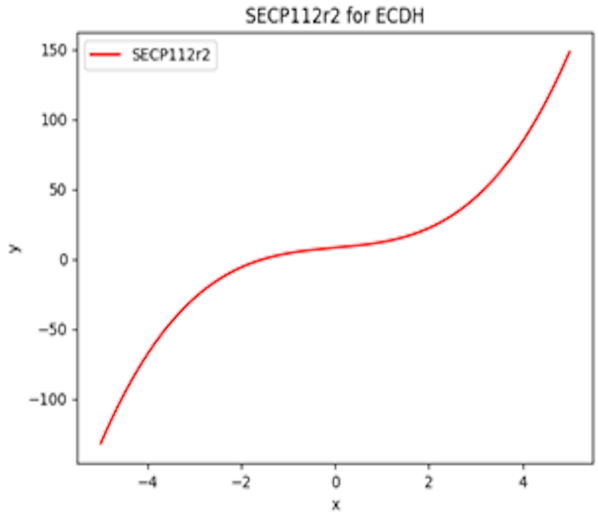


(b) SECP 112r1B ECDH Curve

Fig. 17. SECP 112r1 ECDSA and ECDH curves

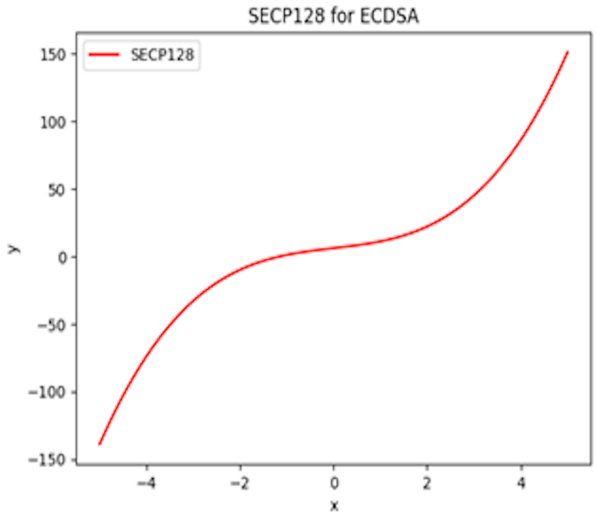


(a) SECP112r2A ECDSA Curve

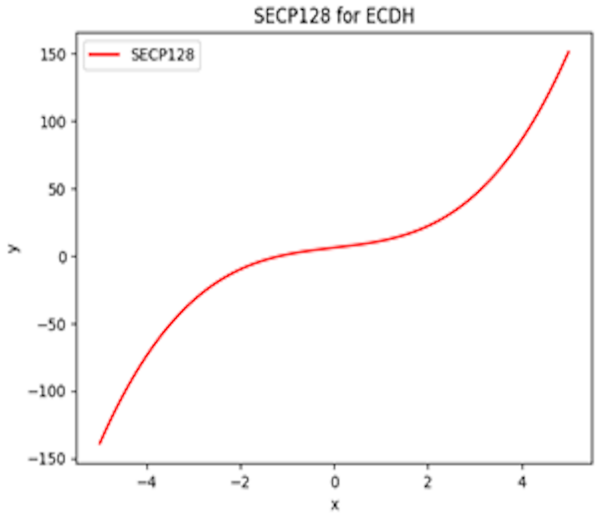


(b) SECP 112r2B ECDH Curve

Fig. 18. SECP112r2 ECDSA and ECDH curves

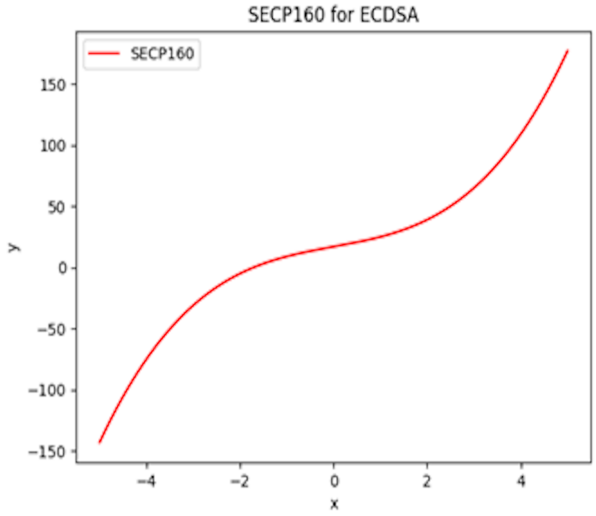


(a) SECP128r1A ECDSA Curve

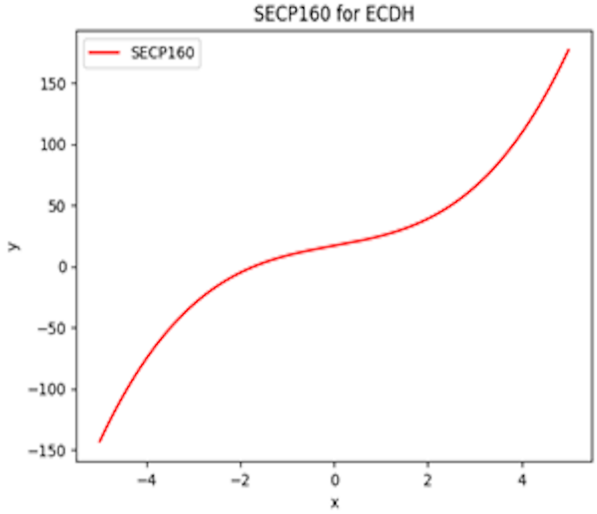


(b) SECP 128r1B ECDH Curve

Fig. 19. SECP128r1 ECDSA and ECDH curves

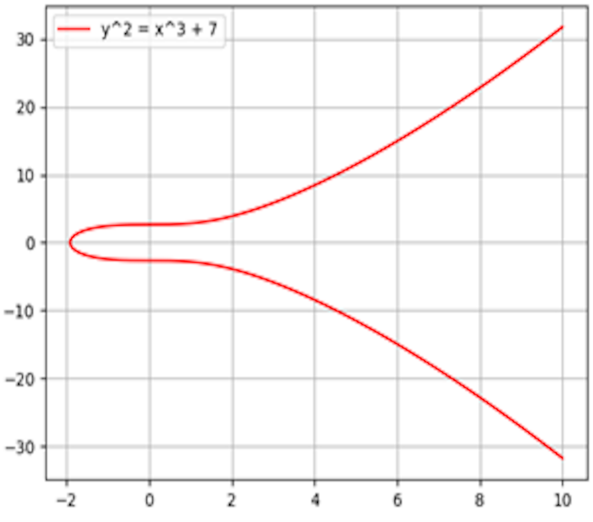


(a) SECP160r1A ECDSA Curve

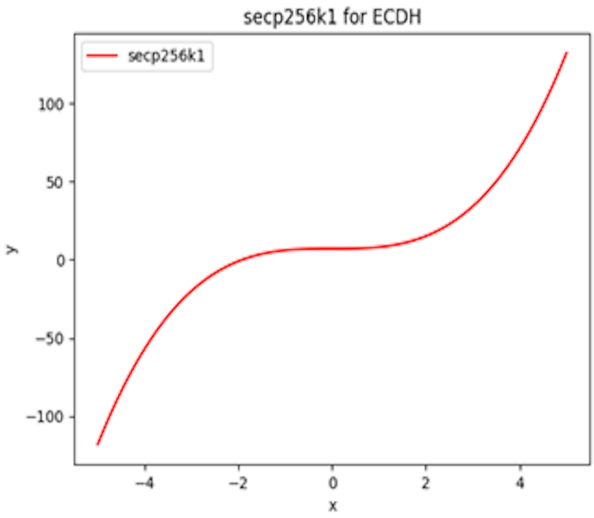


(b) SECP 160r1B ECDH Curve

Fig. 20. SECP160r1 ECDSA and ECDH curves



(a) SECP256k1A ECDSA Curve



(b) SECP256k1B ECDH Curve

Fig. 21. SECP256k1 ECDSA and ECDH curves

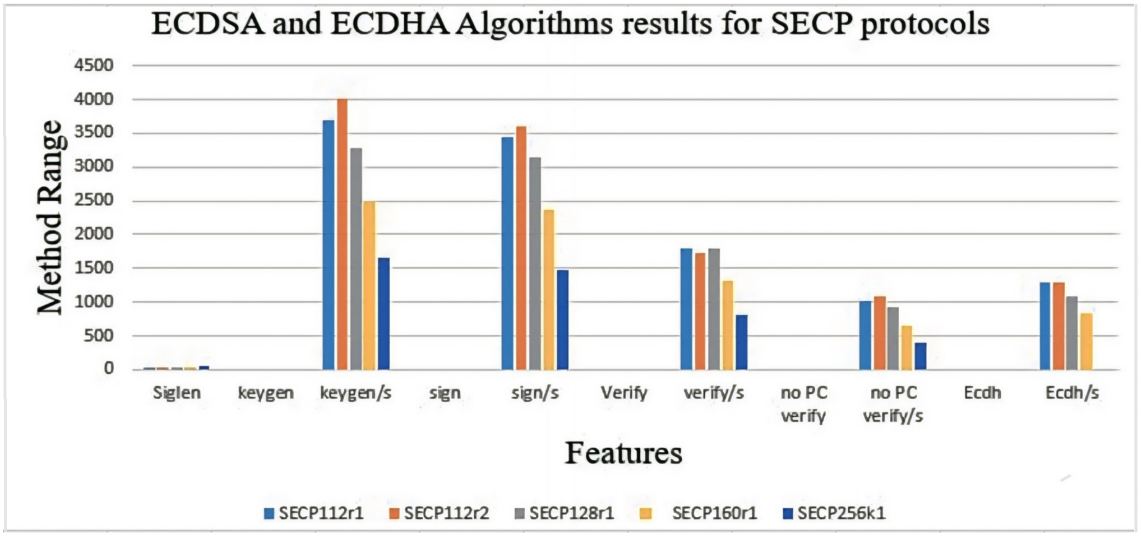


Fig. 22. SECP Protocols comparison for ECDSA and ECDH features

IV. Conclusion

In this research, an extensive assessment and comparison of the ECDSA and ECDHA algorithms were carried out. Thorough examination, both theoretically and experimentally, has illuminated the strengths and weaknesses of the significant techniques involving elliptic curve encryption. The findings indicate that ECDSA and ECDH display distinct performance traits in various coding tasks. ECDSA stands as the preferred option for ensuring data authenticity and integrity, excelling in generating and validating digital signatures. The comparative Evaluation of these techniques emphasizes how performance and security must be carefully balanced. Although ECDSA provides stronger digital signatures, its computational requirements are comparatively larger. On the other hand, ECDH provides effective key exchange but lacks intrinsic signature capabilities. These insights help experts choose the best algorithm based on aspects like computing power, processing speed, and security level for their particular cases. Even with this thorough analysis, elliptic curve cryptography is still a dynamic field with ongoing developments and new problems.

Conflict of interest

The authors have no conflict of relevant interest to this article.

References

[1] V. V. Zubov, "An electronic signature within the digital economy," in *International Scientific Conference GCPMED*, 2019.

[2] N. Saxena and N. S. Chaudhari, "Secure encryption with digital signature approach for short message service," in *2012 World Congress on Information and Communication Technologies*, pp. 803–806, IEEE, 2012.

[3] M. Al-Zubaidie and R. A. Muhajjar, "Integrating trustworthy mechanisms to support data and information security in health sensors," *Procedia Computer Science*, vol. 237, p. 43 – 52, 2024.

[4] P. K. Shukla, A. Aljaedi, P. K. Pareek, A. R. Alharbi, and S. S. Jamal, "Aes based white box cryptography in digital signature verification," *Sensors*, vol. 22, no. 23, p. 9444, 2022.

[5] W. A. Jebbar and M. Al-Zubaidie, "Transaction security and management of blockchain-based smart contracts in e-banking-employing microsegmentation and yellow saddle goatfish," *Mesopotamian Journal of CyberSecurity*, vol. 4, no. 2, p. 1 – 19, 2024.

[6] R. H. Razzaq and M. Al-Zubaidie, "Formulating an advanced security protocol for Internet of medical things based on blockchain and fog computing technologies," *Iraqi Journal for Computer Science and Mathematics*, vol. 5, no. 3, p. 723 – 734, 2024.

[7] G. Uganya and R. Baskar, "Modified elliptic curve cryptography multi-signature scheme to enhance security in cryptocurrency.," *Computer Systems Science & Engineering*, vol. 45, no. 1, 2023.

[8] M. Al-Zubaidie, Z. Zhang, and J. Zhang, "Efficient and secure ECDSA algorithm and its applications: A survey," *International Journal of Communication Networks and Information Security*, vol. 11, no. 1, p. 7 – 35, 2019.

[9] I. Damgård, T. P. Jakobsen, J. B. Nielsen, J. I. Pagter, and M. B. Østergaard, "Fast threshold ECDSA with honest majority," *Journal of Computer Security*, vol. 30, no. 1, pp. 167–196, 2022.

[10] Y. Lindell and A. Nof, "Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1837–1854, 2018.

[11] B. Jana and J. Poray, "A performance analysis on elliptic curve cryptography in network security," in *2016 International Conference on Computer, Electrical & Communication Engineering (ICCECE)*, pp. 1–7, IEEE, 2016.

[12] S. Vasundhara, "Elliptic curve cryptography and diffie-hellman key exchange," *IOSR Journal of Mathematics (IOSR-JM)*, vol. 13, no. 1, pp. 56–61, 2017.

[13] H. M. Al-Mashhadi and M. H. Alabiech, "Symmetric ecc with variable key using chaotic map," *International Journal of Computer Science Issues (IJCSI)*, vol. 14, no. 6, pp. 24–28, 2017.

[14] K. E. Abdullah and N. H. M. Ali, "A secure enhancement for encoding/decoding data using elliptic curve cryptography," *Iraqi Journal of Science*, pp. 189–198, 2018.

[15] M. I. Jabiullah and K. N. Arifa, "An ecsda-based security approach on blockchain for cryptocurrencybased online transactions," *Recent Innovations in Wireless Network Security*, vol. 2, no. 2, pp. 1–12, 2020.

[16] H. B. Mahajan and A. A. Junnarkar, "Smart healthcare system using integrated and lightweight ecc with private blockchain for multimedia medical data processing," *Multimedia Tools and Applications*, vol. 82, no. 28, pp. 44335–44358, 2023.

[17] K. Singh, O. Dib, C. Huyart, and K. Toumi, "A novel credential protocol for protecting personal attributes in blockchain," *Computers & Electrical Engineering*, vol. 83, p. 106586, 2020.

[18] S. Aikins-Bekoe and J. B. Hayfron-Acquah, "Elliptic curve diffie-hellman (ECDH) analogy for secured wireless sensor networks," *International Journal of Computer Applications*, vol. 176, no. 10, pp. 1–8, 2020.

[19] A. Saepulrohman and T. P. Negara, "Implementation of elliptic curve diffie-hellman (ECDH) for encoding messeges becomes a point on the gf (pp)," 2020.

[20] G. Ogunleye and S. Akinsanya, "Elliptic curve cryptography performance evaluation for securing multi-factor systems in a cloud computing environment," *Iraqi Journal of Science*, pp. 3212–3224, 2022.

[21] H. M. Al-Mashhadi and A. K. Ala'a, "Hybrid homomorphic cryptosystem for secure transfer of color image on public cloud," *International journal of computer science and network security*, vol. 18, no. 3, pp. 48–55, 2018.

[22] W. Stallings, "Digital signature algorithms," *Cryptologia*, vol. 37, no. 4, pp. 311–327, 2013.

[23] A. Khalique, K. Singh, and S. Sood, "Implementation of elliptic curve digital signature algorithm," *International journal of computer applications*, vol. 2, no. 2, pp. 21–27, 2010.

[24] O. S. Nagesh, V. S. Naresh, *et al.*, "Comparative analysis of MOD-ECDH algorithm and various algorithms," *International Journal of Industrial Engineering & Production Research*, vol. 31, no. 2, pp. 301–308, 2020.

[25] I. Cherkaoui, "Diffie-hellman multi-challenge using a new lossy trapdoor function construction.," *IAENG International Journal of Applied Mathematics*, vol. 51, no. 3, 2021.

[26] M. S. Khan, D. S. Sakkari, *et al.*, "Security and performance analysis of elliptic curve crypto system using bitcoin curves.," *IAENG International Journal of Computer Science*, vol. 50, no. 2, 2023.

[27] L. Chen, D. Moody, A. Regenscheid, and K. Randall, "Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters," tech. rep., National Institute of Standards and Technology, 2019.

[28] D. Hankerson and A. Menezes, "Elliptic curve cryptography," in *Encyclopedia of Cryptography, Security and Privacy*, pp. 1–2, Springer, 2021.

[29] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 2018.

[30] C. Paar and J. Pelzl, *Understanding cryptography*, vol. 1. Springer, 2010.

[31] M. Koppl, D. Siroshtan, M. Orgon, S. Pocarovsky, A. Bohacik, K. Kuchar, and E. Holasova, "Performance comparison of ECDH and ECDSA," in *2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT)*, pp. 825–829, IEEE, 2021.

[32] M. Qu, "Sec 2: Recommended elliptic curve domain parameters," *Certicom Res., Mississauga, ON, Canada, Tech. Rep. SEC2-Ver-0.6*, 1999.