Volume 10, Issue 7, July – 2025 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

Enhancing 6G Network Security with Elliptic Curve Cryptography

Jury O. Balgoon[1]; Kulud H.Ma Chung[2]; Latifah M.Alharthi[3]

[1,2,3] Dept. of Computer Engineering Taif University, KSA

Publication Date: 2025/09/06

Abstract: With a rapid influx of endeavors into the sixth-generation (6G) wireless realm, network security is becoming of paramount importance. This work investigates strategies to couple Elliptic Curve Cryptography (ECC) with potentially reinforcing the existing 6G security architecture. ECC constitutes a highly secure and efficient key exchange and authentication mechanism, requiring much smaller keys for excellent-level encryption. With such properties, ECC fits very well in the resource-constrained environments envisioned with 6G, such as the Internet of Things (IoT) ecosystem and smart home applications. With the active consideration of performance, scalability, and lightweight communication security, ECC-based strategies will render the required protection against adversarial attacks in high-speed and high-density networks. The strengths of ECC in securing communication, protecting user data, and ensuring privacy in multiple 6G applications have been elucidated in this paper.

Keywords*: 6G Security, 6G Protection, Anomaly Detection, Threat Forecasting.*

How to Cite: Jury O. Balgoon; Kulud H. Ma Chung; Latifah M. Alharthi (2025). Enhancing 6G Network Security with Elliptic Curve Cryptography. *International Journal of Innovative Science and Research Technology,* 10(7), 3894-3901. https://doi.org/10.38124/ijisrt/25jul1664

I. INTRODUCTION

With 5G specs still under construction and coverage not nearing completion, the concept of sixth-generation (6G) mobile communication is pacing in the shadows. The overriding factor driving this change is how telecom networks are already endowed with some level of interconnected intelligence, which is now enhanced by the new AI [1].

The ethical impact of 6 G on wireless connectivity demands high-level security for network operations.

Little literature precisely focuses on the security and privacy aspects of 6G networks, and a set of standard features and standards for 6G is still to be settled. This article emphasizes the importance of further research in this area and outlines the possible security consequences of the foreseen 6G wireless technologies and potential countermeasures.

As the more advanced mobile broadband (FeMBB) brings about extreme data rates, processing will be quite difficult regarding security-related traffic detection of attacks, AI/ML pipeline use, traffic analysis, and ubiquitous encryption. To decrease this inappropriate use, distributed security solutions should come into play because traffic has to be treated locally and dynamically through various operational segments of the network, starting from the edge infrastructure to the core service cloud [2].

Softwarization and virtualization of network services are the most important aspects of developing new generation (5G and 6G) networks, which have been helping to serve various businesses on the same shared physical space and resources. Multi-tenancy, the term used to refer to these possibilities of sharing, enables this.

The use of Internet of Things (IoT) devices increases by leaps and bounds and is expected to rise further. With 6 G knocking at the door and promising to speed things up like never before, lower latency, and an unusual amount of devices connected, the heightened security concern over IoT devices will seem far more justified [4]. Kaspersky, by stating that one troubling research question remains about whether it should come first in protecting IoT devices themselves or networks from IoT device attacks, this issue will not be resolved soon [5].

A project analyzes how Elliptic Curve Cryptography (ECC)—a cryptographic scheme—can be applied to the encryption and decryption of data flow through 6G networks. The ECC works on the concept of curve complexity and also on the algebraic structure of finite elliptic curves. Most of its attributes correspond to the asymmetric parts of cryptosystems, namely key exchange, digital signatures, and encryption[6].

Elliptic Curve Cryptography (ECC) is useful for digital signatures, encryption, and authentication, among other crucial security tasks. ECC creates keys based on

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

properties of the elliptic curve equation rather than the conventional approach of factoring huge prime numbers.

The research's main contribution is the design, prototype, and unmanned protection of an end-to-end data exchange in the 6G-IoT infrastructure against several attacks, such as DDoS attacks using ECC. This study produces the following new and important contributions.

II. RELATED WORKS

The authors of [7] proposed a framework that discerns potentially infected Internet of Things (IoT) devices in a botnet under the detection of malicious traffic at the edge layer of IoT. The analysis is performed by recomputing the threshold of the decision point relevant to traffic classification using Sparsity Representation and Reconstruction Error Threshold methods. The computation of the threshold error only considers benign traffic data, while the training of the machine learning models is performed on an NB-IoT dataset. However, the applicability of this framework has not yet been verified at all levels of the 5G/6G architecture, and its viability in specific settings involving multiple stakeholders has not been evaluated either. Additionally, the authors do not consider classifying 5G and 6G traffic; these, therefore, depend on the validation of the approach.

An entropy-based DDoS detection system for Software-Defined Networking (SDN) attacks is proposed in [8], using a hybrid two-level detection methodology that relies on information entropy combined with deep learning methods. With relatively coarse granularity, the first level does an entropy-based detection mechanism to identify suspicious ports and components. The second level does a fine-level detection of legitimate versus suspect traffic by using a Convolutional Neural Network (CNN) model for packet classification. Ultimately, the controller is responsible for stopping the attack by forcing AI agents to stop it.

The authors have defined a restriction on Subscriber Concealed Identifiers (SUCI) in [9] in the context of cryptography, more so in the post-quantum 6G era. The authors formulated an SUCI for securing SIM cards based on the NIST-declared post-quantum Key Encapsulation Mechanisms (KEM) standard. Solutions are emphasized as potential safeguards against quantum attacks.

Article [10], discussing IoT in the 4G and 5G networks, proposes a method to identify those IoT devices at risk of infection. The attack is mitigated by putting the traffic from these devices in quarantine on a Network Slice (NS) specifically for this purpose. The quarantined traffic is thoroughly examined to determine the status of being malicious. Detection is performed by an application running on the SDN controller; changing numbers of flows in the quarantine, NS, causes the application to produce an updated distrust threshold. In contrast, our approach focuses on the collaboration of the entire infrastructure for protection, while this work focuses on ISPs with no mitigation in the DSP.

III. PROPOSED SOLUTION

Elliptic Curve Cryptography must be integral to the 6 G security architecture to respond to unforeseen demand requirements with respect to speed, low latency, and connectivity of a massive number of devices. However, with considerably smaller key sizes and equally high security compared to traditional systems such as RSA, ECC becomes convenient even in resource-constrained environments peculiar to 6G, particularly in IoT devices, autonomous systems, and real-time AI communications. The implications for ECC technology in the security attributes of 6 G will be enhanced data protection, network performance improvement, and resource conservation.

In a 6G scenario, ECC may be deployed to secure device authentication via methods such as ECDSA, provide fast and efficient protocols for key exchange like ECDH, and enable encryption of sensitive data traveling in ultra-fast networks, with lightweight features offered by ECC itself. Thus, ECC lessens the computational overhead requirements for quick processing and lower power; this is critical for mobile and edge devices operating in a 6G environment.

Security within decentralized systems like blockchain networks and federated AI models expected to underpin many applications in 6G relies on ECC. However, that can be an assured long-term security base by combining ECC with post-quantum cryptographic methods because of the impending quantum computing threat. Choosing an appropriate hardware implementation with standard curve selection must avoid vulnerabilities such as side-channel attacks and ensure global interoperability across 6G infrastructures. Thus, ECC is foundational in building secure, scalable, and future-resilient 6G systems.

The first step to implement ECC for securing 6G networks is to select standard and secure elliptic curves, such as Curve25519 or NIST P-256, for interoperability and hardened cryptographic bases. Each device must obtain a unique ECC key pair at manufacturing or onboarding, ideally through secure hardware modules (such as TPMs or secure enclaves). Under ECDSA use, devices will be authenticated by verifying their digital signatures against a trusted certificate authority (CA) during registration or attaching to a network.

The ECDH key exchange must be implemented in the session layer, allowing the devices to derive shared symmetric keys without exposing their private keys and ensuring confidentiality and integrity in communications. Base stations and end devices must be incorporated with lightweight ECC libraries optimized for mobile and IoT environments, thus consuming as little computational and energy resources as possible. Hybrid encryption models could also be established where the session keys derived from ECC begin/coalesce symmetric encryption, such as AES, to facilitate data transfer.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

ECC operations must also be shielded against side-channel attacks using hardware-level defenses such as constant-time algorithms and fault detection.

IV. EXPERIMENTS AND RESULTS

Our proposed model has the following implementation steps. First, we select standardized and secure elliptic curves, namely Brainpool, NIST P-256, or Curve25519, and we avoid weak or deprecated curves. Second, we securely generate ECC private/public key pairs for device manufacturing or onboarding. Private keys must be kept in secure hardware, such as TPMs, Secure Enclaves, or dedicated chips for cryptography.

Then, we must set up a Trusted Certification Authority (CA) that issues digital certificates binding device identities to their ECC public keys. The devices must trust this CA to validate each other's certificates. Next, authentication is done using ECC, namely ECDSA, where ECDSA signatures are used every time the device enrolls or moves around the 6G network. The network checks the signature against the stored public key to identify the entity.

In addition, ECC Key Exchange (ECDH) is fast and allows secure key exchanges between network nodes and devices. This means symmetric keys are derived for encrypted session operations with AES encryption. Next, those lightweight ECC libraries like WolfSSL or micro-ecc will be targeted for specialized optimization design in 6G hardware, concerning ECC algorithm optimization for hardware and implementing countermeasures against side-channel attacks like constant-time computation next-before.

Secure and Encrypt Data. Messages exchange during a key exchange can also use the symmetric keys to secure signaling messages, data packets, control planes, and user sessions. They can also be applied to secure end-to-end encryption of data in transit.

An ECC private key and an ECC public key are generated for ECC. The private keys in ECC are integers, commonly in the range of 256 bits. The random integer chosen from the range defined on the elliptic curve generates the key in ECC cryptography. The whole set of random integers within the range is a private key in ECC.

Public keys within ECC represent points on a curve as pairs of integer coordinates {x, y}. The unique property of EC points is that they can be compressed to one coordinate + 1 bit (even or odd). Consequently, the compressed public key consists of a 257-bit integer corresponding to a 256-bit ECC private key.

The equation describes the elliptic curves, which are flat algebraic curves made up of all points {x, y}:

$$A x^3 + B x^2 y + C x y^2 + D y^3 + E x^2 + F x y + G y^2 + H x + I y + J = 0$$

The simplified version of elliptic curves used in ECC cryptography, known as the Weierstras form[6], is described as:

$$y^2 = x^3 + ax + b$$

For instance, an elliptic curve of the following shape serves as the basis for the NIST curve secp256k1 (used in Bitcoin):

$y^2 = x^3 + 2$ (the above elliptic curve equation, where $a = 0$ and $b = 2$)

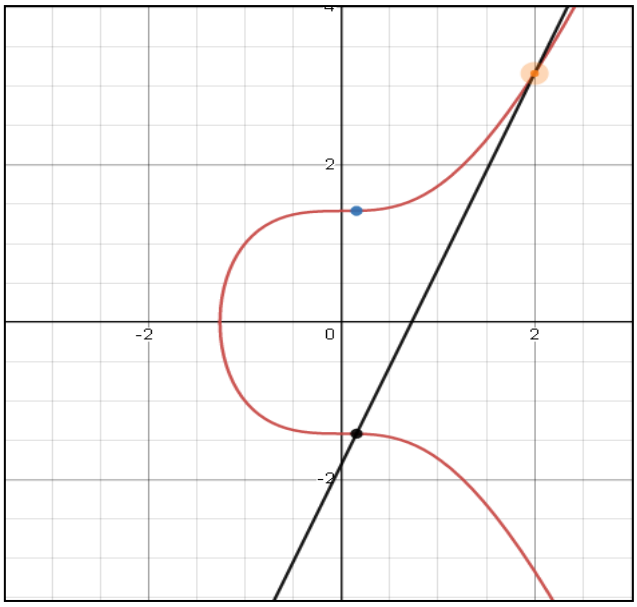

Fig 1 Elliptic Curve Equation

ECC employs elliptic curves over the finite field [11-12] F2m (where the field size is p = 2m) or Fp (where p is prime and p > 3). As a result, the field is a square matrix of size p x p, and the curve's points can only be integer positions inside the field. Every algebraic operation in the same field yields a different point in that field. Thus, the following is the representation of the elliptic curve equation over the finite field Fp:

$$y^2 \equiv x^3 + ax + b \ (mod\ p)$$

For instance, the "Bitcoin curve" (secp256k1) uses an elliptic curve over the finite field F17:

$$y^2 \equiv x^3 + 7 \ (mod\ p)$$

This section describes how to use public-key encryption and decryption based on elliptic curves. Assume that you have an ECC private-public key pair and that you must use it to encrypt and decode data. According to the definition, this rule governs how asymmetric encryption operates. Figure 2 shows that if data is encrypted using a private key, the corresponding public key can later decipher the ciphertext.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

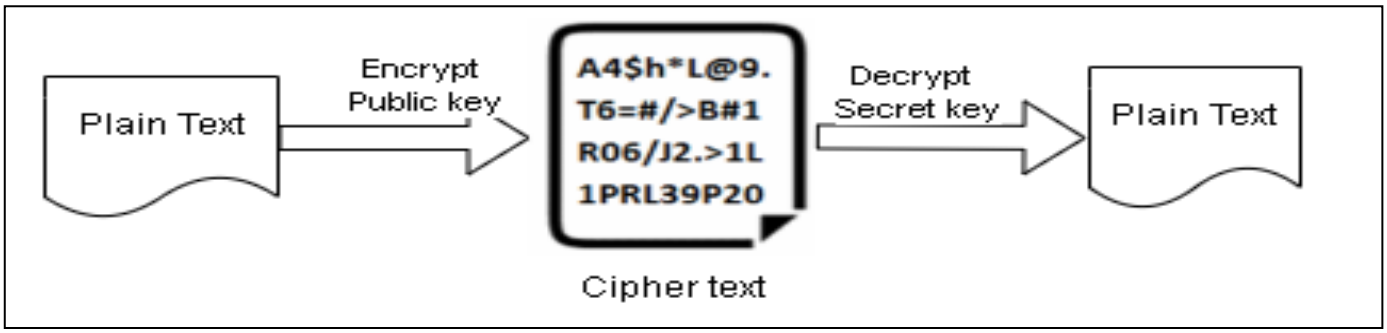

Fig 2 Asymmetric Encryption Process

The RSA cryptosystem can use the aforementioned procedure, however the ECC cannot. The encryption mechanism is not directly provided by elliptic curve cryptography (ECC). In order to create a shared secret key for symmetric data encryption and decryption, this study suggested designing a hybrid encryption system employing the ECDH (Elliptic Curve Diffie–Hellman) key exchange scheme.

Using the tinyec library, the Python code creates an ECC private-public key pair for the recipient of the message (based on the brainpoolP256r1 curve). It then uses the recipient's public key to generate an ephemeral ciphertext public key (for ECDH) and a secret shared key (for encryption). Later, it uses the recipient's private key and the previously generated ephemeral ciphertext public key to generate the same secret shared key (for decryption). This is the output of the code shown above:

- *Private Key:*
0x5c47513f125a733a019060adc831b0e0cbd476dd63724db61b33eec9fa9516f9

- *Public Key:*
0x56f1932b33181ce1da84075e49432806d21debe33964fa13ae139038eaf4d84d0

- *Ciphertext Pubkey:*
0x9de17a915b9e23ab94e88e411cf87351cc800e557432e10c3c91d6dcd62075751

- *Encryption Key:*
0x35e01f2af3d22ee83cce16c0a4f632ae34a2ccd82675073ca5cfee95168f794a1

- *Decryption Key:*
0x35e01f2af3d22ee83cce16c0a4f632ae34a2ccd82675073ca5cfee95168f794a1

The output makes it evident that the decryption key, which is generated from the matching private key, and the encryption key, which is derived from the public key, are identical.

In an integrated encryption scheme, these keys will be utilized for both data encryption and decryption. If you run the code, the result above will differ since ciphertextPrivKey is generated randomly, but the encryption and decryption keys (the ECDH shared secret) will always be the same.

Using a symmetric encryption system such as AES-GCM, the secret key is utilized for symmetric data encryption once it is available. Let's put into practice a fully functional hybrid technique for asymmetric ECC encryption and decryption. The AES authenticated symmetric cipher and the brainpoolP256r1 curve will serve as its foundation.

Using the tinyec library, the preceding example begins by creating an ECC keys pair for the message recipient: pubKey + privKey. Using the hybrid encryption strategy (asymmetric ECC + symmetric AES), these keys will be used to encrypt the message (for example, the user password) and subsequently decrypt it back to its original form.

Next, use the pubKey to encrypt the message. The output will look like this: { ciphertext, nonce, authTag, ciphertextPubKey }. Symmetric AES encryption yields the ciphertext, nonce (random AES initialization vector), and authTag (the encrypted text's MAC code, as determined by the GCM block mode).

In order to retrieve the AES symmetric key during the decryption process, it is also necessary to collect a randomly generated public key, ciphertextPubKey, which will be contained in the encrypted message.

Together with the decryption privateKey, the data generated during encryption—which looks like this: { ciphertext, nonce, authTag, ciphertextPubKey }—is used to decrypt the encrypted communication. The decrypted plaintext message is the end result.

Internally, the encrypt_ECC(msg, pubKey) function computes the symmetric encryption shared ECC key sharedECCKey = ciphertextPrivKey * pubKey after first creating an ephemeral ECC key-pair for the ciphertext. Since this key is an EC point, its x and y coordinates are hashed to convert it to a 256-bit AES secret key (integer).

Lastly, using the 256-bit shared secret key secretKey, the AES cipher (from Pycryptodome) encrypts the message and outputs {ciphertext, nonce, authTag}.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

The symmetric encryption shared ECC key sharedECCKey = privKey * ciphertextPubKey is initially determined internally by the decrypt ECC (encryptedMsg{ciphertext, nonce, authTag, ciphertextPubKey}, privKey) function.

Since it is an EC point, the x and y coordinates of the point were first hashed to convert it to a 256-bit AES secret key. The 256-bit shared secret key secretKey is then used to decrypt the {ciphertext, nonce, authTag} using the AES cipher. The original plaintext message is the output that is generated. The following is the outcome of the code mentioned above:

original msg: b'Text to be encrypted by ECC public key and decrypted by its corresponding ECC private key'

- *Encrypted Msg: {'Ciphertext':*

b'6c0c9051c90e3247c31b5165d1a3d101c4bfd2454ca395e2586b0f3abde2741c8ae0a6b8d027ef8cc4f841dc88037e3c69209354dff8d6c36dd1dccaee1906d063b80cedb50597a5564815eca557caa090acae0c4a1cdcd06f', 'nonce': b'd04a25533676bfdd085c8864d2ecb17e', 'authTag': b'd0514a9709f3d5d20319b15db0532e20',

- *'Ciphertextpubkey':*

'0x5afb37a10e972ad4c8aca946106a1ab93badd70b9dbba74fba19d8d293f0d9221'}

decrypted msg: b'Text to be encrypted by ECC public key and decrypted by its corresponding ECC private key'

In an e-learning environment, the data that is transferred is expressed in JSON format. It can be encrypted and decrypted using the previously described manner and is handled as plain text.

Fog computing is used to solve latency problems that may arise while utilizing ECC cryptography to protect user authentication and e-learning resources. However, over a limited time, fog computing fully synchronizes with the cloud.

- *Comparison Between ECC and RSA:*

One viable asymmetric-key cryptosystem is rival Shamir Adleman (RSA) [13]. The de facto standard for public-key cryptography is established. Its security falls within the challenge of integer factorization. RSA's encryption method is more efficient than its decoding method. Numerous scholars have suggested using the Chinese Remainder Theorem (CRT) to increase the decryption efficiency of RSA. Verma et al. [14] suggested a model to use CRT to speed up the RSA's decryption time. Additionally, they suggested using a matrix with a small order to generate cryptographic keys and huge modulus.

Larger key sizes are necessary for improved and more robust data protection, which puts additional strain on computer systems.

Three sample data inputs of 8 bits, 64 bits, and 256 bits, along with random private keys in accordance with NIST's standard, were used to test RSA and ECC for information security [15]. The Python tests are conducted on an Intel Pentium dual-core CPU running on the Microsoft Windows platform (2.60 GHz, 533 MHz, 1 MB L2 cache) with 8GB DDR4 RAM.

Table 1 Key 8 Bits – Encryption, Decryption and Total Time (in Seconds) [6]

Security Bit Level	Encryption Time (s)		Decryption Time (s)		Total Time	
	ECC	RSA	ECC	RSA	ECC	RSA
80	0.4885	0.0307	1.3267	0.7543	1.8152	0.785
112	2.203	0.0299	1.5863	2.7075	3.7893	2.7375
128	3.8763	0.0305	1.769	6.9409	5.6453	6.9714
144	4.7266	0.0489	2.0022	13.6472	6.7288	13.6962

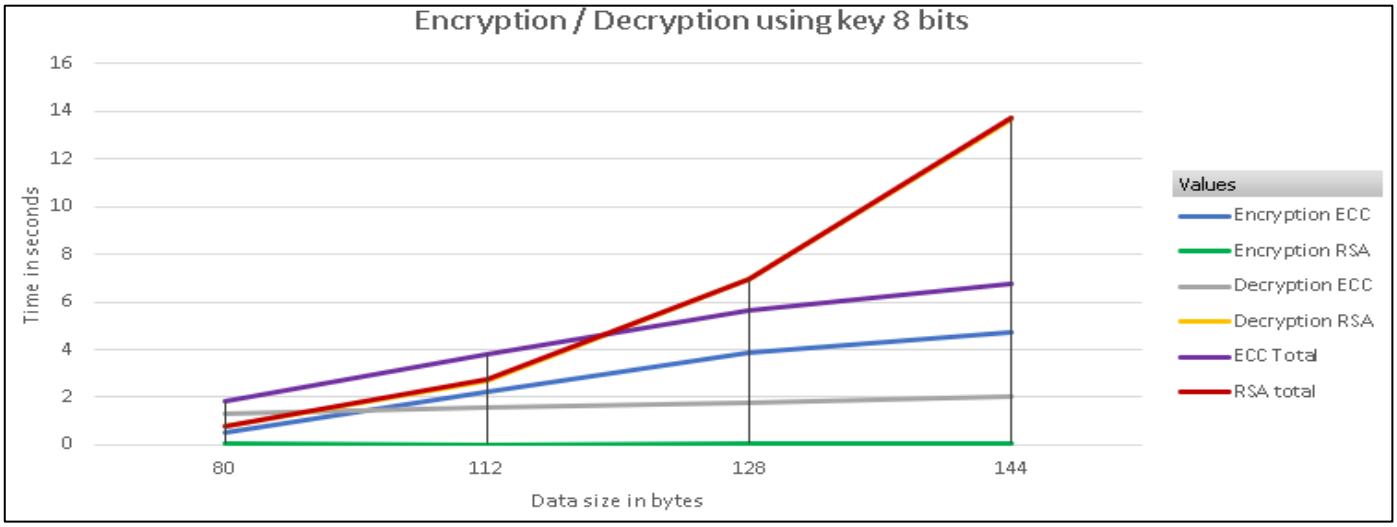

Fig 3 Encryption / Decryption Using Key 8 Bits[6]

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

When utilizing key 8 bits, ECC's encryption time is longer than RSA's in Figure 3, but ECC's decryption time is shorter. Nevertheless, ECC takes less time to encrypt and decrypt than RSA.

Table 2 Key 64 Bits – Encryption, Decryption and Total Time (in Seconds) [6]

Security Bit Level	Encryption Time (s)		Decryption Time (s)		Total Time	
	ECC	RSA	ECC	RSA	ECC	RSA
80	2.1685	0.1366	5.9099	5.5372	8.0784	5.6738
112	9.9855	0.1635	6.9333	20.4108	16.9188	20.5743
128	15.0882	0.1672	7.3584	46.4782	22.4466	46.6454
144	20.2308	0.1385	8.4785	77.7642	28.7093	77.9027

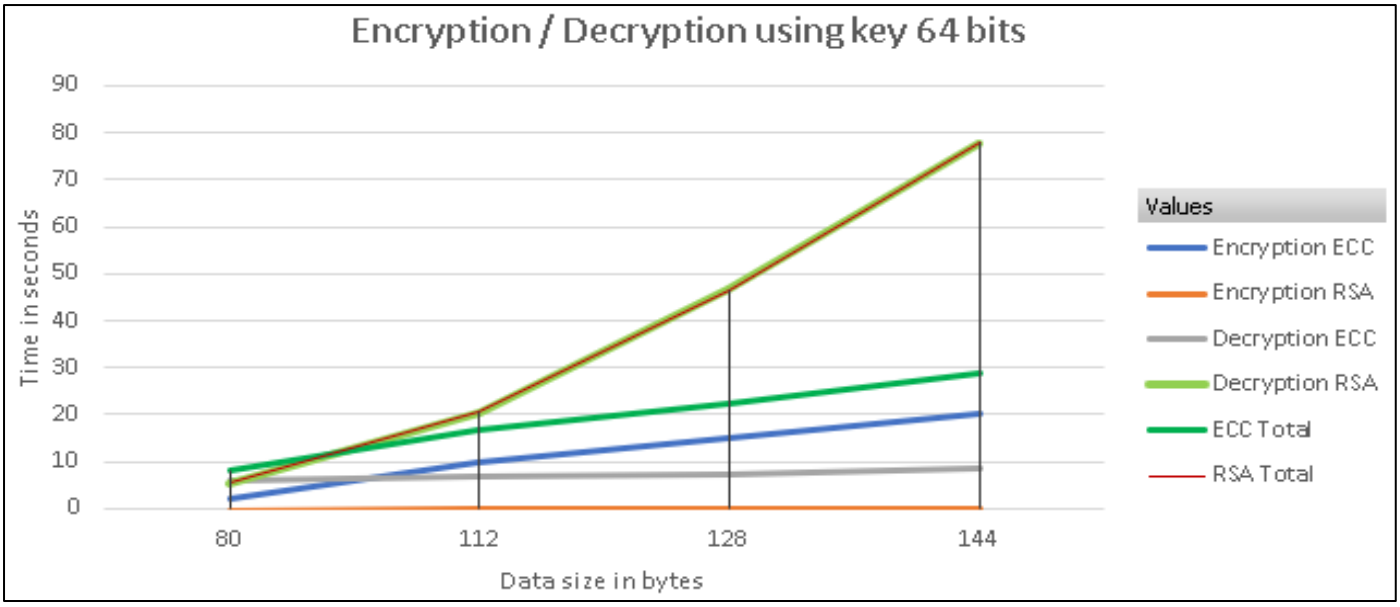

Fig 4 Encryption / Decryption Using Key 64 Bits[6]

Figure 4 shows how to use a 64-bit key for encryption and decryption. RSA encryption takes longer than ECC encryption, but ECC encryption takes less time to decrypt than RSA encryption, which intersects with the overall RSA encryption/decryption time. ECC takes significantly less time overall than RSA.

Table 3 Key 256 Bits – Encryption, Decryption and Total Time (in Seconds) [6]

Security Bit Level	Encryption Time (s)		Decryption Time (s)		Total Time	
	ECC	RSA	ECC	RSA	ECC	RSA
80	7.9240	0.5596	22.8851	19.3177	30.8091	19.8772
112	39.7008	0.5815	26.3331	102.0337	66.0339	102.6153
128	58.4386	0.5611	27.4060	209.6086	85.8446	210.1697
144	77.5034	0.5718	32.1522	311.0649	109.6556	311.6368

V. CONCLUSION

In conclusion, ECC integration can provide an effective and reliable solution against the challenging demands expected to arise in wireless communication shortly in 6G environments. With its small key size, fast processing ability, and battery conservation within devices, ECC will serve as the appropriate design to safeguard extensive device interconnectivity and sensitive information flows projected in 6G environments.

By combining ECC authentication, key exchange, and encryption with secure hardware implementations, robust end-to-end security can be obtained in 6G networks while allowing for ultra-low latency and reliability needed for the next-generation application. However, successful deployment will demand perfect adherence to cryptographic best practices, careful standardization, and continuous security assessments that help alleviate the range of attacks that may arise. Thus, ECC will be a central mast on which the security, scalability, and resilience of a 6G future will hinge.

Securing these networks has become paramount with the rise of 6G networks and their demand for the highest possible connectivity, fastest possible speed, and astute management. The answers are offered by the newly adapted concept of Elliptic Curve Cryptography, which is believed to give strong security assurances while using minimal computational and energy resources. This fits in with the application range of diverse and resource-poor devices that would populate the ecosystems of 6G.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

6G systems can build trust, privacy, and integrity on secure ECC-based authentication, key exchange, lightweight encryption, robust hardware protection. Nevertheless, continuous innovations and monitoring and strict compliance with globally accepted cryptographic standards will remain central to the best defenses against the threats raised, ensuring that ECC will always be a platform for secure 6G deployment in the coming years.

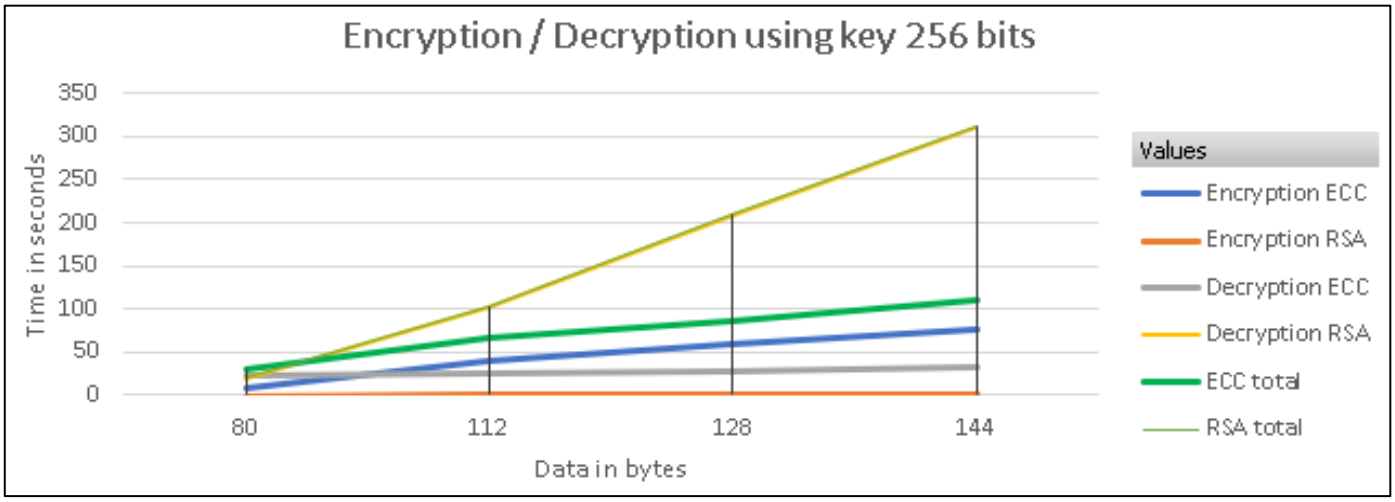

Fig 5 Encryption / Decryption Using Key 256 Bits[6]

In Figure 5, ECC encryption is higher than RSA encryption, but ECC decryption is lower than RSA decryption. ECC encryption and decryption use less time overall than RSA encryption and decryption.

According to earlier findings, ECC is more efficient than RSA. Experiments and the findings show that ECC is slow at encryption but very effective at decryption, while RSA is very effective at encryption but slow at decryption. ECC is more secure and efficient overall than RSA, albeit [15, 16, 17].

However, the encryption/decryption procedure for RSA is longer than for ECC as the key size increases. In the encryption/decryption process, key size matters. When employing a large value key, such as 256 bits, ECC performs better than RSA.

- *Elliptic Curve Cryptography Offers Several Benefits Over RSA Certificates:*

Increased security. Even though RSA is impenetrable right now, experts think ECC will be more resilient to attacks in the future. Therefore, employing ECC could provide you with increased security down the road. Increased effectiveness. Your website may lag if you use huge RSA keys because they require a lot of processing power to encrypt and decrypt data. On the other hand, ECC can scale up more effectively without using significant computer power.

Complete confidentiality. Put simply, this means that even in the event that the private key is compromised, the session keys—which are actually used to encrypt the data transferred between the user and the server—remain safe. If a website is being monitored by outside parties, this could be helpful.

REFERENCES

[1]. C. de Alwis, A. Kalla, Q. V. Pham, P. Kumar, K. Dev, W. J. Hwang, and M. Liyanage, "Survey on 6G Frontiers: Trends, Applications, Requirements, Technologies, and Future Research," IEEE Open Journal of the Communications Society, pp. 1–1, 2021.

[2]. G. Gui, M. Liu, F. Tang, N. Kato, and F. Adachi, "6G: Opening new horizons for integration of comfort, security, and intelligence," IEEE Wireless Communications, 2020.

[3]. McKinsey&Company, The road to 5G: The inevitable growth of infrastructure cost, https://www.mckinsey.com/industries/technology-media and-telecommunications/our-insights/the-road-to-5g-the-inevitable-growth-of infrastructure-cost.

[4]. Lionel Sujay Vailshery, Global IoT market size, https://www.statista.com/ statistics/976313/global-iot-market-size/.

[5]. Nikolay Pankov, Protect networked IoT devices or protect the network from IoT devices? https://www.kaspersky.com/blog/rsa2021-dangerous-iot/40161/.

[6]. El Sayed Amer, M.S.M., El Hefnawy, N., Mohamed Abdual-Kader, H. (2024). "Enhance Fog-Based E-learning System Security Using Elliptic Curve Cryptography (ECC) and SQL Database." International Conference on Innovative Computing and Communications. ICICC 2023. Lecture Notes in Networks and Systems, vol 731. Springer, Singapore. https://doi.org/10.1007/978-981-99-4071-4_34.

[7]. Christos Tzagkarakis, Nikolaos Petroulakis, Sotiris Ioannidis, Botnet attack detection at the IoT edge based on sparse representation, in: 2019 Global IoT Summit, GIoTS, 2019, pp. 1–6, http://dx.doi.org/10.1109/GIOTS.2019.8766388.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25jul1664

[8]. Ying Liu, Ting Zhi, Ming Shen, Lu Wang, Yikun Li, Ming Wan, Software-defined DDoS detection with information entropy analysis and optimized deep learning, Future Gener. Comput. Syst. 129 (2022) 99–114, http://dx.doi.org/10.1016/j. future.2021.11.009.
[9]. Ulitzsch, V.Q.; Park, S.; Marzougui, S.; Seifert, J.-P. A Post-Quantum Secure Subscription Concealed Identifier for 6G. In Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks, San Antonio, TX, USA, 16–19 May 2022; pp. 157–168.
[10]. David Candal-Ventureira, Pablo Fondo-Ferreiro, Felipe Gil-Castiñeira, Fran cisco Castaño, Quarantining malicious IoT devices in intelligent sliced mobile networks, Sensors (Basel, Switzerland) 20 (2020) http://dx.doi.org/10.3390/ s20185054.
[11]. W. H. Bussey (1910) "Tables of Galois fields of order < 1000", Bulletin of the American Mathematical Society 16(4): 188–206, doi:10.1090/S0002-9904-1910-01888-7.
[12]. Mullen, Gary L.; Mummert, Carl (2007), Finite Fields and Applications I, Student Mathematical Library (AMS), ISBN 978-0-8218-4418-2.