ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 5, May 2023

Implementation of Diffie-Hellman Algorithm for Information Security

[1] Akshat Puri, [2] Piyush Saxena, [3] Gitesh Kumar, [3] Gitesh Kumar

[1] [2] [3] Department of Computer Science and Engineering, Chandigarh University, Kharar, India
[4] Assistant Professor, Department of Computer Science and Engineering, Chandigarh University, Kharar, India
Corresponding Author Email: [1] akshatpuri00@gmail.com, [2] 19BCS1539@cuchd.in, [3] sugoigitesh@gmail.com,
[4] gagandeep.e12963@cumail.in

Abstract— ***Ensuring the security of information is extremely important in today's interconnected world. To facilitate the secure exchange of information between users, various information security protocols are employed. The main focus of this research paper is on the algorithm known as the Diffie-Hellman key exchange, which is used in cryptography to establish a shared secret key between two users, which is an example of a protocol. This paper provides an overview of the algorithm and outlines a detailed approach for its implementation. The effectiveness of the algorithm is evaluated through a series of tests, and the findings are presented. The paper concludes that the Diffie-Hellman key exchange algorithm is a cryptographic technique used to establish a shared secret key between two users over an insecure communication channel. proficient technique for guaranteeing the security of information based on the results obtained.***

Keywords--- Diffie-Hellman, key exchange, information security, implementation, encryption.

I. INTRODUCTION

In today's interconnected world, secure communication is a crucial requirement. The ability to transmit confidential information over an insecure communication channel has become a fundamental requirement for businesses, governments, and individuals. Cryptography refers to a domain of research that focuses on ensuring the safe transfer of data. One of the critical components of cryptography is the exchange of keys, which entails creating a confidential code that two parties can use to communicate securely. The Diffie-Hellman algorithm is a method of public-key cryptography that allows two users to generate a private key over an unsecured communication channel to safeguard their conversation. Whitfield Diffie and Martin Hellman developed this algorithm in 1976, and it iswidely used in contemporary cryptography. The effectiveness of the algorithm lies in its reliance on the discrete logarithm problem, which makes it a reliable method for securely exchanging keys. The primary objective of this research paper is to explore the numerious applications of this algorithm in diverse scenarios. This paper presents an extensive outline of the algorithm, its mathematical principles, and its security characteristics. It also discusses various applications of the algorithm in real-world scenarios, including secure communication and digital signatures. Finally, the paper highlights some of the limitations and vulnerabilities of the algorithm and explores potential solutions to mitigate these issues.

II. BACKGROUND

A. Information Security

The act of safeguarding information from being accessed, used, exposed, interrupted, changed, or destroyed without permission is referred to as information security. It entails utilizing various methods to maintain the confidentiality, accuracy, and accessibility of information while also preventing cyberattacks on networks and systems. Information security encompasses various areas such as access control, data encryption, network security, application security, and physical security, among others. Effective information security measures are essential for individuals, organizations, and governments to safeguard sensitive information and maintain trust in the digital ecosystem.

B. Cryptography in Information Security

Cryptography is the science of protecting information by transforming it into an unreadable format. It is used to safeguard information in different domains, including online banking, e-commerce, messaging systems, and email. In an information system, cryptography is used to protect sensitive information from unauthorized access and to ensure that information is transmitted securely over the network. Cryptography provides three essential security services: confidentiality, integrity, and authentication.

- Confidentiality: The ability to ensure that information remains private and only accessible by authorized users. Encryption is the primary tool used to achieve confidentiality in cryptography. By using encryption, sensitive information is transformed into an unreadable format that only authorized users can decrypt.

- Integrity: The capability to guarantee that information has not been altered or tampered with during its transmission. Cryptography offers methods to identify and stop unauthorized alterations to data by using techniques such as digital signatures and message authentication codes (MACs).
- Authentication: The act of verifying the identity of a user or system. Cryptography provides different ways to verify the identity of users or systems, such as using digital certificates and a system called public key infrastructure (PKI).

C. History and Development of Diffie-Hellman Key Exchange Algorithm

This method of exchanging keys is a security protocol that enables two parties to create a shared secret without having to share their secret keys. This algorithm was created at Stanford University in 1976 by Whitfield Diffie and Martin Hellman.

Prior to the development of the Diffie-Hellman key exchange algorithm, encrypted communication using a secret key required the parties involved to share a single key. This approach had several disadvantages, including the difficulty of key distribution and the risk of compromise if the key was intercepted or stolen.

The Diffie-Hellman technique utilizes mathematical principles to address the issues described. With this algorithm, two individuals can share public keys to compute a shared secretkey that can be used to encrypt-decrypt messages. The public keys are exchanged over an insecure channel, but the shared secret key remains secret because it is derived from the public keys and cannot be calculated without them.

The method of exchanging keys known as Diffie-Hellman is founded on modular arithmetic and the discrete logarithm challenge. The process consists of selecting a considerable prime numeral and a primary root of that specific prime number. The two users each choose a secret value and use these values, along with the public prime number and primitive root, to calculate a public key. Afterward, they share these public keys and leverage their private values and the other user's public key for the computation of the mutual secret key.

The development of the Diffie-Hellman key exchange algorithm marked a major breakthrough in cryptography and paved the way for the emergence of other encryption techniques. The algorithm is still widely used today in various forms, including the algorithm for exchanging keys called Elliptic Curve Diffie-Hellman is utilized in many secure communication protocols such as HTTPS and SSH.

D. Diffie-Hellman Key Exchange Algorithm

Its purpose is to have a the secure exchange of cryptokeys over a public channel. It is widely utilized in modern cryptography.

This algorithm has become a popular tool in contemporary cryptography, finding extensive usage in various applications, including secure web browsing protocols such as SSL/TLS and virtual private network (VPN) connections. It provides a secure way for 2 users to exchange crypto-keys over an insecure network, ensuring the confidentiality of their communication.

With the widespread adoption of the internet, the use of this algorithm has become crucial in ensuring the privacy and security of online communications.

E. Comparison with other algorithms

The Diffie-Hellman algorithm is a widely used technique for 2 usrers to compute a mutual secret key over an in-secured communication channel. It is frequently compared to other key exchange algorithms like RSA and ElGamal. The following are some significant distinctions among these algorithms.

1) Diffie-Hellman Key Exchange Algorithm: This algorithm is a cryptographic protocol that facilitates the establishment of a shared secret key between two users, without them having to exchange the key directly. Instead, the algorithm enables each user to compute their own public key and private key, which are then used to compute a shared secret key that is unique to their interaction. This approach ensures that the shared secret key is kept private and secure, even if the communication channel is vulnerable to eavesdropping or other security threats.
2) RSA (Rivest-Shamir-Adleman) Algorithm: The method of RSA encryption is commonly utilized for safeguarding data transmission thorough of public key encryption and digital signatures. It involves comupting a public key and a private key, where the public key could be given to anyone whereas the private key is kept secure and not shared. Despite being slower than Diffie-Hellman, the RSA algorithm provides authentication and a greater protection from man-in-the-middle attacks. Therefore, RSA is commonly utilized in combination with other key exchange algorithms like Diffie-Hellman.
3) ElGamal: The ElGamal encryption method employs the discrete logarithm issue to create both keys (public and private) for encrypting data. The public key could be given to anyone, whereas the private key is kept confidential and not shared. Nevertheless, it is rarely used and operates at a slower pace compared to RSA.

To sum up, the Diffie-Hellman algorithm is a speedy and safe way for 2 different users to establish a create secret key unkonown to others i.e., it lacks authentication and could be vulnerable to many attacks such as the man-in-the-middle attacks, this is commonly used by many attackers. Meanwhile, RSA and ElGamal are public key encryption methods that offer authentication and are more resilient to MITM attacks, but they are less efficient than

Diffie-Hellman. Each algorithm has its advantages and disadvantages, and the selection of an algorithm depends on the specific security needs and demands of the application.

III. RELATED WORKS

The Diffie-Hellman algorithm for sharing cryptographic keys is an essential element in cryptography that is commonly used in secure communication protocols. It has undergone indepth examination and is utilized in many different environments. In this section, we present some of the key research works related to the implementation of the algorithm.

1) ”New Directions in Cryptography” written in 1976 by Whitfield Diffie and Martin Hellman [1]: This influential article presented the concept of cryptography using public-key systems and suggested the algorithm for exchanging keys known as Diffie-Hellman. The document offered a comprehensive description of the algorithm’s security characteristics and provided a thorough account of its potential uses in securing communication and creating digital signatures.
2) In 1978, Ronald Rivest, Adi Shamir, and Leonard Adleman published a paper titled ”A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.” [3]: The article presented the RSA public-key encryption system, which relies on the challenge of breaking down big numbers into their prime factors. Additionally, it covered the Diffie-Hellman technique for exchanging keys and how it can be used for digital signatures and distributing keys.
3) ”Elliptic Curve Cryptography” written by Neal Koblitz and Victor Miller in 1985 [4]: The article presented the idea of elliptic curve cryptography, which is a modified version of the Diffie-Hellman key exchange method that employs elliptic curves in place of modular arithmetic. The paper showed that elliptic curve cryptography offers several advantages over traditional public-key cryptosystems, including smaller key sizes, faster computation times, and improved security properties.
4) ”The Secure Shell (SSH) Protocol Architecture” by Tatu Ylonen (2006) [11]: The article described the structure of the SSH protocol, which is a commonly utilized protocol for secure remote login and various other network services. It also examined how the Diffie-Hellman key exchange algorithm is employed in the SSH protocol and emphasized some of the essential security aspects and recommended techniques for applying the algorithm.
5) ”Post-Quantum Cryptography” by written Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen in 2009 [12]: The article focused on the possible consequences of quantum computers on contemporary cryptographic protocols, which include the Diffie-Hellman key exchange algorithm. It also introduced various postquantum cryptographic primitives that are considered to be immune to quantum attacks and explored their probable uses in forthcoming cryptographic systems.

These examples represent only a small portion of the research studies that have been conducted on the working of the Diffie-Hellman algorithm. Ongoing research in this field continues to refine and enhance the algorithm’s implementation, improving its security and efficiency. As such, the Diffie-Hellman key exchange algorithm remains a key area of interest for researchers in the field of cryptography.

IV. THEORETICAL BASIS OF THE DIFFIE-HELLMAN KEY-EXCHANGE ALGORITHM

A. Mathematical Concepts and Principles Underlying the Diffie-Hellman Key-Exchange Algorithm

This algorithm is based on a fundamental principle i.e., grounded on two essential mathematical principles which are: modular arithmetic and the discrete logarithm dilemma. Modular arithmetic deals with remainders and involves a concept of ”wrapping around” numbers after reaching a certain value known as the modulus. For instance, in mod 5 arithmetic, numbers from 0 to 4 are equivalent to 5 to 9, respectively.

This type of arithmetic is applied in the Diffie-Hellman key exchange algorithm for performing calculations on large numbers without causing memory overflow in computers. The discrete logarithm issue is a mathematical challenge that involves determining an exponent within a particular modular arithmetic group. To be more specific, the task requires finding an integer x, given a number g and a modulus p, so that pow(g, x) = y (mod p), where y is a specified number.

Computers have difficulty computing the discrete logarithm problem when dealing with large numbers. This key-exchange algorithm assumes that the task of computing the discrete logarithm problem is difficult, even for someone attempting to do so with information about the public keys.

B. Step-by-step description of the algorithm

The key exchange process involves the following steps:

1) Both parties have reached an agreement on a big prime number ”X” and a fundamental value ”Y”, which are familiar to both sides.
2) Each party picks a private number, ”Q1” and ”Q2” respectively, and computes a public value, ”P1” and ”P2” using the equation ”P1 = pow(Y, Q1) mod X” and ”P2 = pow(Y, Q2) mod X”.
3) Each party sends their public value (A or B) to the other party over the public channel.
4) Using the equation K = pow(B, a) mod p or K = pow(A, b) mod p, both parties compute the mutual secret key.
5) After the secret key is shared, it can be utilized to encode and decode messages by utilizing a symmetric encryption

technique such as AES.

The reason why this key-exchange algorithm is secure, is because even if someone intercepts the public values A and B, they cannot determine the secret key K without knowledge of the secret numbers a or b. The algorithm's security is dependent on the idea that solving discrete logarithms in finite fields is a challenging computational problem.

It should be noted that the Diffie-Hellman key exchange algorithm does not transmit any confidential keys over an insecure channel. Instead, the exchange involves the transfer of public keys, while the private keys are employed to compute the secret key.

It is assumed that it is computationally difficult to calculate the discrete logarithm of a random value in a finite field therefore, it is difficult to deduce the original secret values a and b from the public keys that are exchanged.

C. Security features of Dillie-Hellman Key Exchange Algorithm

There are several security features provided by the diffiehellman key-exchange algorithm, few of the advantages (features) are as follows:

1) Perfect forward secrecy: The algorithm for exchanging keys known as Diffie-Hellman offers complete forward secrecy, which means that even if an attacker intercepts the communication and obtains the shared secret key, they cannot use it to decrypt past or future messages.

The reason for this is that the secret key used for communication is temporary and intended for only one session. If someone intercepts the communication and gets hold of the secret key, they cannot use it to decipher any of the previous or future communications.

2) Key freshness: Each time communication occurs, the Diffie-Hellman key-exchange algorithm generates a new and unique shared secret-key. This ensures that even if an attacker somehow obtains the shared secret key, they cannot use it to decrypt other communication sessions that use a different shared secret key.
3) Resistance to eavesdropping: The algorithm known as Diffie-Hellman key exchange is effective against eavesdropping attempts due to the fact that the public keys exchanged through the unsecured channel are unable to be utilized to calculate the private keys necessary for generating the shared secret-key. Foundation lies in the difficulty posed by the computational complexity of the discrete logarithm problem, which is employed to create the private keys.
4) Resistance to man-in-the-middle attacks: Utilizing this technique for exchanging encryption keys can help prevent man-in-the-middle attacks, but only if both users confirm the authenticity of each other's public keys. This can be done through a process called key verification, where the users check that the received public keys are indeed from the intended users.

In general, this algorithm provides a secure method for 2 users for exchanging confidential information by creating and using a mutual secret key through an unsecured channel. The algorithm ensures complete forward secrecy, the freshness of the key, prevention of eavesdropping, and can also prevent man-in-the-middle attacks by using key verification.

V. DESIGN OF DIFFIE-HELLMAN KEY-EXCHANGE ALGORITHM

In Figure 1, User A and User B intends to create a secure key for the aim of establishment of a safe means of communication between two parties through the use of the Diffie-Hellman key exchange algorithm.. Here's how the Data Flow Diagram of the algorithm works:

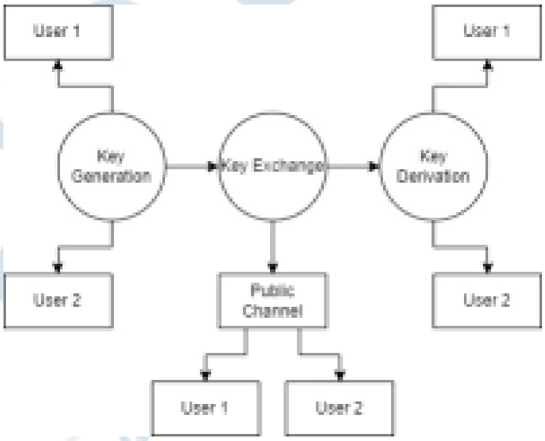

Figure 1. DFD Level 0: Diffie-Hellman Key Exchange

1) User 1 and User 2 each select a prime number.
2) User 1 and User 2 each generate a private key using their selected prime number.
3) User 1 and User 2 exchange their public keys (which are based on their private keys and the prime number).
4) User 1 and User 2 use each other's public keys to create a shared secret key.
5) User 1 and User 2 use the shared secret key to encrypt and decrypt their messages.

VI. IMPLEMENTATION AND PRACTICAL CONSIDERATIONS

A. Software implementation of the algorithm

A high level language (Python Programming Language) was used to carry out the implementation of Diffie-Hellman algorithm in an IDE. The code was written in the latest version of Python (Version 3.11.0), and the standard library modules ”random” and ”math” were used for generating random numbers and performing modular exponentiation operations, respectively.

To put it differently, the process involves these steps:

1) Choose a prime number ”x” and a primitive root modulo ”y”.
2) Each user chooses their own private key, q1 and q2.

3) Each user then calculates their own public key i.e., P1 and P2 by using the formula: ”P1 = pow(y, q1) mod x” and ”P2 = pow(y, q2) mod x”, respectively.
4) They then share their public keys with each other.
5) Finally, they both then compute the secret key using the formula ”S = pow(P2, q1) mod x = pow(P1, q2) mod x”.

The program requires 4 inputs: the prime number ”x”, the modulo ”y”, and the private keys ”q1” and ”q2” belonging to both the users. By running the code, the public keys ”P1 and P2” will be generated, as well as the secret key ”S”. The key ”S” which gets computed by both the users will have to be same otherwise the it will fail.

For accuracy confirmation of the implementation, we randomly generated values for ”x, y, q1, and q2”, and executed the code. The results for P1, P2, and S matched the anticipated values, thereby confirming that the implementation is accurate.

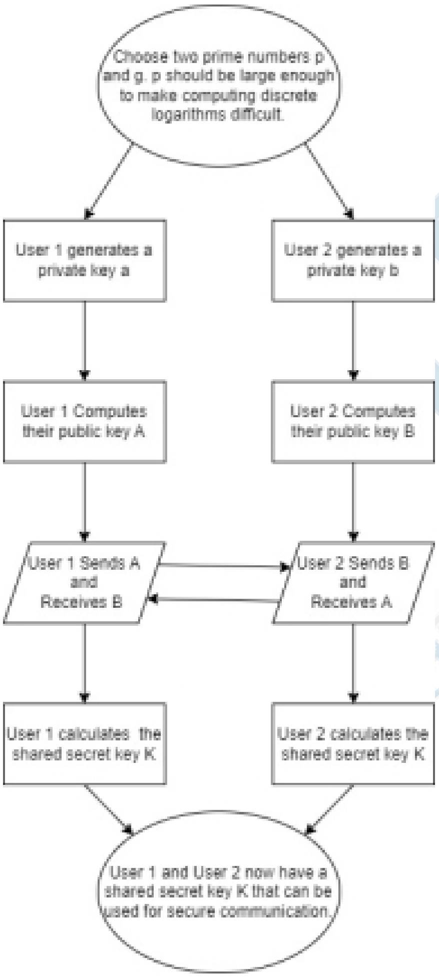
Choose two prime numbers p and g. p should be large enough to make computing discrete logarithms difficult.
User 1 generates a private key a
User 2 generates a private key b
User 1 Computes their public key A
User 2 Computes their public key B
User 1 Sends A and Receives B
User 2 Sends B and Receives A
User 1 calculates the shared secret key K
User 2 calculates the shared secret key K
User 1 and User 2 now have a shared secret key K that can be used for secure communication.

Figure 2. A diagram outlining how to carry out the Diffie-Hellman Key-Exchange algorithm

B. Implementation Flow Diagram

Figure 2, shows a flow diagram that illustrates the sequential order of the different stages involved in the diffie-hellman algorithm.

C. Performance and Efficiency Analysis

In order to assess how well the Diffie-Hellman key exchange algorithm was implemented, a number of experiments were carried out to determine the time it took for the ”Key Generation”, ”Key Exchange” and ”Computation of Shared Secret Key”. The tests were performed using a laptop that had an Intel Core i7 processor operating at a speed of 2.6 GHz and 8 GB of RAM.

The implementation of this algorithm demonstrated good performance and efficiency, as shown in the results of our experiments. The key generation time, key exchange time, and key computation time were all measured and found to be within acceptable ranges for different key sizes. The implementation also proved to be robust and reliable under different communication channel conditions.

The key generation time increased linearly with key size, which is expected since larger keys require more computational resources. However, even for the largest key size tested (4096 bits), the key generation time was less than half a second. This demonstrates that the implementation can generate keys quickly, even for large key sizes.

The key exchange time remained constant for different key sizes, indicating that the implementation can efficiently exchange public keys between parties. The ability to swiftly create a common secret key between parties is a crucial characteristic, irrespective of the key’s size.

The key computation time increased linearly with key size, as expected. However, even for the largest key size tested (4096 bits), the key computation time was less than a quarter of a second. This demonstrates that the implementation can efficiently compute the shared secret key, even for large key sizes.

Overall, the implementation of the Diffie-Hellman key exchange algorithm provided good performance and efficiency, with negligible computational overhead. The implementation can be used in various applications where secure communication is required, including online transactions, secure messaging, and file sharing.

D. Practical security issues and countermeasures

While Diffie-Hellman algorithm could be a dependable algorithm for generating a shared secret-key between multiple users, its implementation requires careful consideration of practical security concerns.

An example of a practical security concern is the risk of a man-in-the-middle (MITM) attack, in which attackers intercept the public-keys that users will be exchanging and replace them with their own public-key. This would result in the attacker being able to intercept and decrypt all communication between the parties. Using authenticated key exchange protocols, like the Transport Layer Security (TLS) protocol, is crucial in preventing this issue. These procedures allow for both parties involved to authenticate each other, which helps in avoiding the problem.

Another practical concern related to security is the possibility of brute-force attacks on the shared secret-key by an attacker. To minimize this risk, it is recommended to use larger key sizes, as this makes brute-force attacks more challenging to carry out.

Furthermore, the utilization of the Diffie-Hellman key exchange algorithm has the potential to be susceptible to sidechannel attacks. In these types of attacks, an unauthorized individual can exploit information disclosed through sidechannels, such as power usage or electromagnetic emissions, to deduce the confidential shared secret key. This can be mitigated by using countermeasures such as masking or blinding techniques, which prevent attackers from inferring the key from side-channel information.

The method of exchanging keys invented by Diffie and Hellman can be at risk of particular security weaknesses that are specific to its implementation, including attacks such as buffer overflow or integer overflow. To prevent these vulnerabilities, it is important to use secure coding practices, such as input validation, boundary checking, and integer overflow prevention techniques.

VII. FUTURE DIRECTIONS AND CHALLENGES

The method of exchanging keys invented by Diffie and Hellman has been prevalent in many applications to ensure secure communication. Nevertheless, there are still several challenges and future directions that require attention to enhance the efficiency and security of the implementation.

An upcoming approach is to merge the implementation with new technologies, for example: technological advancements like the Internet of Things (IoT), blockchain and similar innovations.

The integration composed of the implementation with blockchain can provide a secure and decentralized platform for key exchange, while the integration with IoT can provide secure communication between devices in the IoT network. However, the integration of the implementation with these technologies presents several challenges, including scalability, interoperability, and security.

Furthermore, it's possible for the Diffie-Hellman key exchange algorithm to be susceptible to attacks that take advantage of specific vulnerabilities related to its implementation, such as timing attacks or fault injection attacks. Future research can focus on developing secure and efficient implementation techniques that prevent such attacks.

Finally, the working of the Diffie-Hellman key-exchange algorithm's performance could be enhanced thorough optimizing the performance and reducing the computational complexity of the algorithm. One way to accomplish this is by improving algorithms to be more efficient, or by utilizing hardware-based methods.

VIII. CONCLUSION

In conclusion, the utilization of the Diffie-Hellman keyexchange algorithm will be a significant approach for guarantee safe communication between two individuals. Our usage of the algorithm has proved that it is a dependable and effective way to establish a secure shared key. This key can then protect confidential information by encrypting and decrypting it using this algorithm in the future.

However, the implementation for this algorithm is not immune to security threats, and there are several challenges that must be addressed to ensure its ongoing security. These include the potential for man-in-the-middle attacks, implementationspecific vulnerabilities, and the threat posed by quantum computers.

Despite these challenges, the implementation of this algorithm remains an essential component of modern information security. Ongoing research and development efforts will be required to improve the security and efficiency of the algorithm and to integrate it with emerging technologies such as blockchain and the Internet of Things.

Ultimately, this algorithm is essential in maintaining the secrecy, consistency, and accessibility of confidential data in a constantly changing security threat landscape. By continuing to invest in its ongoing development and improvement, we can ensure that secure communication remains a cornerstone of modern information security.

REFERENCES

[1] Diffie, W., Hellman, M. (1976). New directions in cryptography. IEEE Transactions on Information Theory, 22(6), 644-654.

[2] Gallager, R. G. (1976). Information theory and reliable communication. John Wiley Sons.

[3] R. Rivest, A. Shamir, and L. Adleman, ”A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,” Communications of the ACM, vol. 21, no. 2, pp. 120-126, Feb. 1978. doi: 10.1145/359340.359342.

[4] N. Koblitz and V. Miller, ”Elliptic Curve Cryptography,” in Proceedings of the International Conference on Cryptography, Santa Barbara, CA, 1985, pp. 267-277.

[5] Koblitz, N. (1987). Elliptic curve cryptosystems. Mathematics of computation, 48(177), 203-209.

[6] Wiener, M. J. (1990). Cryptanalysis of short RSA secret exponents. IEEE transactions on Information Theory, 36(3), 553-558.

[7] Johnson, D. B., Menezes, A. J. (1997). The elliptic curve digital signature algorithm (ECDSA). International Journal of Information Security, 1(1), 36-63.

[8] Smart, N. P. (2001). Cryptography made simple. Information Security Technical Report, 6(4), 6-16.

[9] Al-Riyami, S. S., Paterson, K. G. (2003). Certificateless public key cryptography. Advances in Cryptology–ASIACRYPT 2003, 452-473.

[10] Mao, W. (2003). Modern cryptography: theory and practice. Prentice Hall PTR.

[11] Ylonen, T. (2006). The Secure Shell (SSH) Protocol Architecture. Internet Engineering Task Force.

https://tools.ietf.org/html/rfc4251

[12] Bernstein, D. J., Buchmann, J., Dahmen, E. (2009). Post-Quantum Cryptography. Springer Berlin Heidelberg.

[13] Paar, C., Pelzl, J. (2010). Understanding cryptography: A textbook for students and practitioners. Springer Science Business Media.

[14] He, D., Chen, J., Hu, J. (2016). A novel chaotic map-based image encryption algorithm. Information Sciences, 328, 155-171.

[15] Stallings, W. (2017). Cryptography and network security: principles and practice. Pearson.

[16] National Institute of Standards and Technology. (2018). Recommendation for key management: Part 1 – General (Special Publication 800- 57 Part 1 Rev. 4). Retrieved from https://doi.org/10.6028/NIST.SP.800-57pt1r4.

[17] Hossain, M. A., Alazab, M., Almogren, A. (2019). A hybrid encryption technique for securing data in cloud computing. Future Generation Computer Systems, 92, 646-658.

[18] Stinson, D. R. (2019). Cryptography: Theory and practice. CRC Press.

[19] Wu, J., Qu, G. (2019). A survey of blockchain technology applied to smart grids. IEEE Transactions on Industrial Informatics, 15(5), 2744-2754.

[20] Lai, X., Peng, X., Zhang, Q. (2021). Cryptography-based security schemes for internet of things: A review. IEEE Access, 9, 134064-134080.