FIT2093 Assignment 1:

Investigating an Application of Cryptography

Topic:

Cryptography in Blockchain

Team Name: T05G01

Tutorial: 05

Team Members:

Chong Ming Sheng (32202792)

Stephen Ng Sze Joo (32204221)

Table of Contents

Introduction

Blockchain refers to a shared ledger technology validating ownership using a peer-to-peer network. The makeup of the word comes from the word ‘block’ referring to the collection of records and ‘chain’ referring to using cryptography to link these blocks stored as a list (Aste, Tasca, & Di Matteo, 2017). The goals of blockchain is to decentralise and make consensus transparent with security and immutability. This blockchain's potential lies in its ability to record asset ownerships, contracts, copyrights, credit or digital identification via a decentralised platform away from the normal governance structure. This is achieved via a consensus protocol which determines a quorum of nodes having access to the data (with varying permission levels) in a blockchain network to validate an accurate order in which the new records are added to the shared ledger in which transactions are created, executed or modified based on the consensus protocol (Mingxiao, Xiaofeng, Zhe, Xiangwei, & Qijun, 2017). In addition to that, a one-way cryptographic hash functions are used to produce an immutable historic record validated by the peers adding a complexity for the data to be tampered with. Hence, we would like to look at how hash functions, asymmetric and symmetric encryption, and digital signatures which are all cryptography techniques, are used within the blockchain system as well, what it is, how it is applied, it’s usability and performance, security vulnerabilities that come as a result of the protocols used as well as a detailed analysis of what we discovered. The cryptography techniques that is hash functions and digital signature carry the main security goal of Integrity and Authenticity respectively towards the blockchain network, maintaining it’s immutable quality as well as maintaining the authenticity of transactions between nodes within the network. Refer to Figure 1.0 for a visual representation of what we will be talking about. Among the resources which we used the most are as listed below:

1. Comparative Analysis of Secured Hash Algorithms for Blockchain Technology and Internet of Things by Parmar, M., & Jit, H. (2021).

2. EC-ElGamal and Genetic Algorithm-Based Enhancement for Lightweight Scalable Blockchain in IoT Domain from Guruprakash, J., & Koppu, S. (2020).

3. Minerva: The curse of ECDSA nonces : Systematic analysis of lattice attacks on noisy leakage of bit-length of ECDSA nonces by Jancar, J., Sedlacek, V., Svenda, P., & Sys, M. (2020).

4. Significance of Elliptic Curve Cryptography in Blockchain IoT with Comparative Analysis of RSA Algorithm by Yadav, A. K. (2021).

5. Comparative study on hash functions for lightweight blockchain in Internet of Things (IoT) by Alfrhan, A., Moulahi, T., & Alabdulatif, A. (2021).

Application Protocol/System Description

Types of Cryptographic Mechanisms

Among the types of cryptographic techniques used in blockchain are hash functions, digital signatures and encryption techniques. Hash functions are algebraic models which take in an input of some arbitrary size and produce corresponding resultant fixed sized output that is known as a hash. A hash function should return hashes that are unique to each input data. In the context of blockchains, each block/transaction on the chain is linked together and hashed to maintain the immutable characteristic of the blockchain (Konashevych & Poblet, n.d.). The immutable ledger of a distributed network is maintain by the following example, if a block X being fed into a hashing algorithm outputs the has XH and we want to add block Y to the chain, we will have to hash XH + block Y + some ephemeral value (nonce) to produce the hash YH and for the attacker to alter transaction of X, which can lead to double spending attack, they will need to solve the proof of work (POW) of X and find a matching block to replace Y which is highly improbable as we add more and more blocks to the chain as a single small change to one block on the blockchain takes on a completely distinct hash string that will be produced hence maintaining the integrity of the blockchain (Dilhara, n.d.). This cryptography technique achieves the security goal of integrity making it such that transactions that have been made are immutable.

Next, we want to touch on digital signatures in the application of blockchain technology. Digital signatures in its simplest form, is a signature attached to a message which authenticates that the message originates from a given sender (Subramanya & Yi, 2006). In the application of blockchain systems, each transaction on the blockchain is signed by the sender and broadcasted to all the nodes within the network to verify using its corresponding public key attached to the senders address, after which the transaction is added onto the shared ledger via hashing . Currently, the most commonly used signature scheme is the Elliptical Curve Digital Signature Algorithm (ECDSA) and Edwards Curve Digital Signature (EdDSA), both of which applies the discrete logarithm problem in the form of a curve and both of this signature scheme take after the Digital Signature Algorithm(DSA) (Dilhara, n.d.). As a result, this application of cryptography ensures the integrity for each transaction and helps maintain the non-repudiation property of the sender such that the authenticity of each transaction can be verified.

Lastly, we want to talk about encryption techniques applied within blockchain systems. As we have mentioned above, hash functions and digital signatures help support the integrity and nonrepudiation characteristics of the network, however privacy and confidentiality of transactions are not preserved. Hence, some sources have proposed privacy preserving architecture with a hierarchical structure enabling different individuals to access the data based on certain attributes, that is the attribute based encryption scheme (Rahulamathavan, Phan, Rajarajan, Misra, & Kondoz, 2017). However, in our paper we will be talking about encryption techniques that we have already learned, and how it applies to blockchain systems.

Threat/Attack Model

With the storage of vast amounts of user data, blockchain platforms have attracted tremendous attention and become one of the main targets among the attackers/hackers in order to make changes to transactions in the blockchain network. In this case, the application of hash functions in the blockchain technology is designed to protect against changes to transactions on the blockchain by chaining the transactions together as mentioned in the previous section (Khanal et al., 2022). In the case where a blockchain platform is hacked, an attacker could not easily change the actual value of the user data even if the hash function being used is known by them (Khanal et al., 2022). This is due to the preimage resistance property of the one-way hash functions where given a hash value of input data, it is considered computationally infeasible to get back the original data. Hence, hash functions are important to address any possibility of consenting versioning providing the immutable characteristic of the shared ledger such that the integrity of the transactions are preserved.

Similar to hashing, application of encryption in blockchain is mainly aimed to protect against user privacy as well as data security, which is related to the confidentiality security goal. Ghazal, Hasan, Abdullah, Bakar, and Al Hamadi(2022) emphasised on the effectiveness of encryption in blockchain in making the access to encrypted data difficult and infeasible for unauthorised users, and therefore the security threat of sensitive data leakage can be highly protected. Since encryption is done by converting and scrambling the ordinary data into a code that is highly impossible to read, the genuine receivers or attackers must have the proper key which is private or secret key in order to decrypt the scrambled messages. Thus, encryption in blockchain is applied to secure user personal data from unauthorised access.

Additionally, as a node on the blockchain, the authenticity of genuine nodes within a network is important to make sure transactions that are made are valid. In other words, one of the security threats that blockchain is designed to protect against is the impersonation of unauthorised parties to get benefits from innocent sources where attackers pretend to be the authenticated users in order to utilise their identities for fraudulent activities (Monrat, Schelen, & Andersson, 2019). Some valid examples would be fraudulent transactions such as illegal credit card usage without the real cardholders' knowledge, phishing and scammed transactions in Bitcoin. Application of digital signatures is designed to purposely address the user authenticity issues by verifying the users on both sides of the transaction to ensure that transactions are genuinely made as well as received by authorised users (Monrat, Schelen, & Andersson, 2019).

Likewise, digital signatures can also be used for checking integrity of data. Another security threat that digital signature is designed to protect against is the modification of original data by an attacker in the middle of transmission without being detected by the recipient. Upon receiving the data from the sender, the recipient must first verify the authenticity of data and signer by computing the value with the use of sender's public key via a specific formula. If and only if this particular computed value is found to be the same as the message received from the sender, then the recipient would choose to accept the message. In this case, the integrity of data can be checked as the recipient could probably tell if the original data from the sender has been modified by someone else in the middle of transmission (if the particular computed value is not the same as received message). Knowing the integrity of data is crucial to prevent scams in blockchain.

Take Bitcoin for example, before a payee agrees on accepting the bitcoin transfer from the payer, they must first validate the payer's signature and check if it is from the genuine party that they are looking for, in order to verify the chain of ownership which helps to prevent themselves from being the victims of scams in blockchain. Also, even if the fraudulent activities occur, digital signatures can well assure that neither the sender nor recipient involved in the particular transaction can deny their availability of having processed the information, as the proof of delivery and identity of sender are provided by the digital signatures (Fang et al., 2020). Overall, digital signatures are helpful for validating and verifying the authentication of transactions in blockchain, which greatly reduces the security threats related to authenticity and non-repudiation issues.

Unusual Impacts

The most prevalent consensus protocols among all which have been introduced to blockchain is called the Proof-of-Work consensus (Han, Foutris, & Kotselidis, 2019). This decentralised consensus mechanism demands numerous network members to squander resources and efforts in solving infeasible and complex mathematical problems based on the application of hash functions in blockchain. Scaling the blockchain worldwide, Proof of Work (PoW) systems targets to employ additional individual blocks to the blockchain and transforms the consensus problem to a competition, which is related to the cryptographic hash functions (Han, Foutris, & Kotselidis, 2019). This process is called "crypto mining", which simply means solving the hard computational problem and retrieving the correct value to satisfy the blockchain's hash functions.

Since multiple processing elements like ASICs, FPGAs and GPUs are essentially needed to support the escalating operational requirements in the processes of crypto mining, study found that crypto mining has direct adverse impacts on the environment for instance the increment of global power consumption as well as scalability challenges (Han, Foutris, & Kotselidis, 2019). Another study by Benetton, Compiani, and Morse (2021) discovered that based on the calculation of counterfactual electricity bills, the USA had additionally spent about one billion US dollars due to the demand-for-electricity consequences by crypto mining.

In short, it is explained that despite the high-security property of hash functions used in blockchain which is dominant for user privacy and data protection, it indirectly leads to negative consequences on the environment as extensively high computational power is required especially when it comes to crypto mining as mentioned above.

On top of that, there is also an unusual impact invited by the encryption in which more modern and new hardware approaches might be required for the implementation of power consuming and heavy computational functions to ensure the requirements of current network speed and higher security demand are met (Sklavos, 2010). Provided that cipher components used for encryption and decryption are usually involved in large and complex algebraic as well as arithmetic modifications, their implementations certainly allocate huge system resources, which is considered inappropriate for hardware implementation (Sklavos, 2010). Notwithstanding, software solutions are not favoured as they are found incapable of meeting the high-performance-and-speed standards for mobile communications and portable devices (Sklavos, 2010).

In order to tackle the hardware issue mentioned above, some suggested building a custom-designed hardware which is specifically tailored and designed for one specific encryption process (Sklavos, 2010). In actual fact, this again raises another issue where the development of such special and particular hardware does not economically worth the enormous investment in its design and testing, especially if it is not aimed to be produced on a large scale to the market.

Application Protocol/System Performance & Usability Results

Hash Functions

Comparison of the performance and effectiveness between various hash functions used in the blockchain technology is done based on their hashing speed and execution time. The hashing speed is affected by the size of input data into the hash function, whereas execution time is affected by the size of the file (Parmar & Jit, 2021).

According to Parmar and Jit (2021), the larger the sequence of input data, the smaller the speed of hashing; the larger the file size, the longer the execution time taken by the hash function.

In summary, based on Figure 2.0, we can conclude that among the four hashing algorithms, on average SHA-1 is the fastest in speed response in both scenarios where the sequence of input data is small and large, with 708.3 milliseconds and 909.3 milliseconds for small and long sequences respectively. Next, based on Figure 2.1, it is depicted that the SHA-1 hash algorithm again takes the shortest time to execute its hashing function for large-size files. In contrast, both SHA-2 and SHA-3 families consisting of variants SHA-256 and SHA-512 are found to perform slower than SHA-1 in hashing as illustrated above.

Thus, it is clearly visualised that hash functions have great usability and flexibility as different hash functions have different security measures and perhaps additional unique features that are designed to fit specific blockchain platforms.

Hence, different blockchain platforms are provided with multiple options of hash functions that are found to be perfect for their respective blockchain systems.

Digital Signatures

Nowadays, there are several famous digital signature algorithms applied in blockchain technology such as RSA (Rivest-Shamir-Adleman) Algorithm, DSA (Digital Signature Algorithm), and ECC (Elliptic Curve Cryptography) (Basha, Veesam, Ammannamma, Navudu, & Subrahmanyam, 2021). ECC includes ECDSA (Elliptic Curve Digital Signature Algorithm) that is used by Ethereum and Bitcoin network and also EdDSA (Edwards-curve Digital Signature Algorithm) that is used by Cardano (Guruprakash & Koppu, 2021), observe Figure 2.2.

The comparison of encryption speed and time complexity between RSA, DSA and ECC algorithms is illustrated in Figure 2.3 and Figure 2.4.

As illustrated in both Figure 2.3 and Figure 2.4, the ECC algorithm takes the shortest time for encryption and at the same time it has the lowest time complexity compared to RSA and DSA algorithms.

Due to the evolution of digital technology, rapid advancement and technological requirements, RSA has gradually become obsolete and transitioned to the more advanced ECC-based digital signature algorithm namely ECDSA. Its superiority helps offer improved security by using smaller keys as well as less operational space (Guruprakash & Koppu, 2021). However, in order to further enhance ECDSA, there exists EdDSA that is proven to perform better in terms of execution time than ECDSA in blockchain-based applications.

The comparison of performance between the two digital signature algorithms used in blockchain systems, namely ECDSA and EdDSA, in terms of their time consumption and number of processes taken is shown as in Figure 2.5 and Figure 2.6.

Figure 2.5 tells that EdDSA has lower time complexity and performs faster than ECDSA due to its lesser computational process for single digital signing. Also, Figure 2.6 tells that the overall performance of EdDSA on average is better than ECDSA as fewer operations are required for single digital signing, due to its less complex working nature if compared to that of ECDSA (Guruprakash & Koppu, 2021).

Encryptions

Both symmetric and asymmetric encryption techniques can be applied in blockchain technology to perform data encryption in order to ensure the security of data.

Nevertheless, asymmetric encryption algorithms such as RSA and ECC are mainly and most commonly applied in the blockchain technology due to their simple key distribution management (Nguyen & Wu, 2018). To elaborate, two asymmetric ciphers are needed for encryption and decryption, which are the sender's private key and the receiver's public key. Therefore, in order to solve the key distribution problem introduced by symmetric encryption such as AES, asymmetric encryption is widely used for data encryption in blockchain.

However, if we are to compare the speed of both symmetric and asymmetric encryptions, it is depicted that symmetric encryption actually has a hugely higher speed, which is almost three times faster, in performing encryption and decryption of messages, if compared to that of asymmetric encryption as shown in Figure 2.7 below. In addition, this statement is supported by Abroshan (2021), who claimed that more generation time is required for asymmetric encryption algorithms which cause them to have lower encryption and decryption speed if compared to that of symmetric encryption algorithms. This is because asymmetric encryption requires two asymmetric keys instead of one as in the symmetric encryption.

Application Protocol/System Security Vulnerabilities or Security Analysis Results

Hash Functions

Based on the appropriation of hash functions in Blockchain networks, we come to the 3 basic ways to evaluate the security of a hash function used within blockchains that is algorithm strength, hash output length and quantum computing.

Firstly, we want to evaluate the algorithm strength that is how collision resistant a hash function is. We would like to turn our attention to SHA-256 and SHA-512 which are the only SHA-2 functions with original designs whereas the rest are modifications of these 2 models (Alfrhan, Moulahi, & Alabdulatif, 2021). SHA-256 functions on 32-bits words and processes 512-bit message blocks to 256-bit digest with a 128-bit prevention against collision whereas the SHA-512 functions on 64-bits words and processes 1024-bits blocks, and outputs a digest of 512 bits with 256-bit security against collision as seen below via the evaluation of hash functions in cycle count in Figure 3.0. We see that SHA-512 outperforms SHA-256 in terms of the number of bits prevention against collision.

Next we want to evaluate the output length of hash functions. As stated by Parmar & Jit (2021), the longer the output length of a hash function the safer it is against brute force search attacks. Based on the Figure 3.1 below we can see that SHA -2 -512 having a larger output length compared to SHA -2 -256 eventually leading to a larger search space than the possible outputs of the function making it collision proof (Parmar & Jit, 2021.).

Lastly, we would like to look at the fault tolerance resource counts for Grover search which is a form of quantum preimage attack. A study estimating the cost of generic quantum preimage attacks on SHA-2 -256. Currently, it takes an estimated 1.27 x 10^{44} T-count, 3.76 x 10^{43} T-depth and $2^{153.8}$ surface code cycles needed to run a brute force Grover search on a fault-tolerant surface code based architecture which would take a total of $2^{12.6}$ logical quantum-bit (qubit) (Amy et al., 2017). Currently, SHA-256 (and by assumption SHA-512) is still secure towards quantum preimage attacks as the largest quantum computer's processor today sits at 433 qubit (IBM, 2022).

Hash functions used in the practical setting of blockchains carry very little security risk and vulnerabilities which in the first place is the reason it is used to give the ledger its immutable property. This is a given as the 3 characteristics of a strong hash function which is the pre-image resistance, second pre-image resistance and collision resistance are present due to the large amount of possible hashes that is $2^{256.}$ For the hash function SHA-256 which is the most rudimentary hash function used within blockchains.

Digital Signatures and Encryption Techniques

Once again we know how important digital signatures play in authenticating transactions and verifying transactions by signing via private keys and verifying using shared keys making sure transactions come from the valid user. Hence, we would like to access the 2 most commonly used public-key cryptography techniques used in blockchains that are RSA and ECC. Firstly, we would like to access it from the perspective of the key length as the longer the key length the more resistant it is against brute force attacks (Yadav, 2021). Overall, from the Figure 3.2 below from a study comparing ECC and RSA in terms of encryption and decryption time yields the result showing that ECC requires less key size to achieve a certain security level as compared to RSA, also showing that ECC encrypts slower but is more efficient in decryption as opposed to RSA which encrypts faster but decrypt slower, hence showing us that ECC is more secure than RSA (Yadav, 2021).

However for a more realistic example, let's compare EdDSA and ECDSA which are both used in current blockchain digital signatures standard. From Figure 3.3 what we see is that using the rho method, the complexity of the method on different curve types and their security level (J & Koppu, 2022). The rho complexity here helps us find out the computational power it takes to solve the discrete logarithm problem using the Pollard's rho algorithm. By analysing this further we come to see that Weierstrass curve is stronger than Edwards curve at any of the security levels of 128, 192, and 256 bits meaning that given any key size, ECDSA outperforms EdDSA of it's same size as it is more resistance towards attacks that is a variant of the rho method.

However, with ECDSA we meet a problem whereby the Digital Signature process requires something known as the nonce which is a random scalar which is used in the key generation process. This may be an issue depending on what libraries and what devices the digital signature algorithm is running on which may render the digital signature generation process vulnerable (Jancar, Sedlacek, Svenda, & Sys, 2020). Additionally, the physical platform which the Digital Signature Algorithm (DSA) runs on can be short circuited and the algebraic properties of the DSA can be used to reveal the private key used in the DSA generation process (Joye & Tunstall, 2012). We shall discuss this in detail in the critical analysis section of the report about how this is accomplished.

Critical Analysis of Results

Hash Functions

Provided that SHA-1 is faster in encryption than SHA-2 and SHA-3 given a large input data as well as large file size as mentioned earlier in the "Performance and Usability Results" section above, SHA-1 is considered more cryptographically broken and vulnerable to attacks (Maliberan, 2019). The fast execution time in SHA-1 hash algorithm is due to its shorter and simpler hash value if compared to SHA-2 and SHA-3, which is also the main weakness of this algorithm (Maliberan, 2019). It was found that by using brute force and rainbow table techniques, SHA-1 with short hash value can be cracked easily. A research study was carried out to attempt a collision attack on SHA-1 algorithm and result showed that SHA-1 hash function is no longer secure in either data transfer or login authentication, as the computation on a 64-GPU took only ten days to successfully crack the SHA-1 hash algorithm due to its short sequence of hash value (Maliberan, 2019).

Additionally, the faster speed in SHA-1 could be due to the smaller output bits produced by the hash function, and definitely smaller output bits will lead to lower security level of the hash function as longer output bits can have more possible combinations which is more difficult for attackers to attack by brute force (Parmar & Jit, 2021).

Based on Figure 3.1, we know that both SHA-2-512 and SHA-3-512 hash algorithms have the highest security level among all the other hash algorithms due to their largest number of output bits, whereas SHA-1 hash algorithms have a relatively small output bit size. However, the large number of output bits indeed increases the latency of SHA-2 and SHA-3 which causes them to have slower speed than SHA-1 when performing hashing.

Further improvement to enhance the practicality of hashing functions could be proposed by developing a concurrent error tolerant hashing algorithm with the integration of more compact hardware, for example with the application of SHA-3 hashing function. This suggestion is supported by Chandran and Manuel (2016), who asserted that the inclusion of comprehensive and exhaustive error detection and correction schemes is crucial to enhance the efficiency of blockchain systems. Plus, the hardware latency should be further minimised in order to diminish delays in blockchain systems, for instance during the transactions.

On top of that, the energy consumption and memory requirements of hash functions should be appropriately optimised. At the same time of fulfilling the ongoing demand of improved security in cryptography applications, we must bear in mind that both the power or energy consumption and implementation requirements may be negatively affected by the utilisation of standards with higher security levels (Rubayya & Resmi, 2014). This is significantly important to be taken into consideration as transactions could be made extremely convenient if they could be done via smaller devices such as cell phones, and therefore these devices may not have the capability to sustain high power implementations due to their relatively low battery capacities.

Looking over to the security analysis of hash functions we would like to turn our attention over to the security of hash functions used in blockchain technology, it was claimed that the hash functions should satisfy the hiding property, despite the traditional security criteria of typical hash functions like SHA-1. Based on the Rogaway-Shrimpton's theoretical framework and its symbolic definitions, research discovered that the most commonly-used hash functions in blockchain like SHA-256 (variant of SHA-2) do fulfil the hiding criterion and at the same time they are considered to be preimage-resistant (Wang, Duan, & Zhu, 2018), where preimage is the plaintext as the data input of a hash function in order to compute a hash value. The combination of these criteria makes the widely-applied hash functions such as SHA-256 in a blockchain system to be extremely difficult to break due to their increased collision resistance, and therefore they are considered safe to be used for consensus processes like Proof of Work (PoW) to approve transactions.

Proofs of the "Pre and Hid" theorems were done via construction of a mathematical model and Las Vegas-type probability algorithm as shown in Figure 4.0. The results obtained via the probability function given in Figure 4.1 revealed that it is almost impossible for an adversary to attack the hash function in polynomial time and there also proved to have a negligible advantage of attacking the particular hash function, which brings to the conclusion that the hash function such as SHA-256 is secure in the sense of Hid.

Additionally, we would like to take into consideration properties that are important to a cryptographic hash function. Firstly, preimage resistance means that it is computationally infeasible to trace the input which hashes to the output of a hash function. Second preimage resistance, means that it is computation infeasible to find any input which has the same output as any specified input and collision resistance means that it is computationally infeasible to find any 2 distinct inputs which has the same output (Rogaway & Shrimpton, 2004).

From the three properties mentioned above, we will use sensitivity that is to see given that a single input bit was changed how much of the output bits are affected. Hence by comparing the performance of different hash functions on different input sizes and observing the throughput to indicate the amount of data processed per second in MB/sec unit with the Sensitivity Matrix which indicates the percentage of change of the output of hashing a random string j^{th} when the single input bit at location i is flipped the results are tabulated as in Figure 4.2, Figure 4.3 and Figure 4.4 (Pillai, Hou, Biswas, & Bui, 2022).

Overall, what we observe is that MD4 and MD5 have better performance scores the hash performance of MD4 and MD5 for input sizes of 8192 bits block size are 759.64 and 485.47 MB/s respectively as compared to the 4381.8, 165.02 and 263.63 MB/s that is the performance of SHA-1, SHA-256 and SHA-512 respectively. However, we observe a tradeoff between security and performance as SHA-256 and SHA-512 having better overall ideal sensitivity scores closer to the ideal at 50% and with the lowest standard deviation overall that is 0.094 and 0.07 as opposed to the 0.154 and 0.128 standard deviation of MD4 and MD5's sensitivity matrices respectively (Pillai, Hou, Biswas, & Bui, 2022). In total, what we see here is that there is a balance between processing speed, that is the performance of a hashing algorithm and the overall security of a hash function ; a faster hash would produce an overall less secure hash.

Digital Signatures

As mentioned earlier in the “Performance and Usability Results” section above, we know that the speed of encryption and time complexity of the ECC algorithm outperforms the RSA algorithm. So, now we have a look at the comparison between the key size used for ECC and RSA to better visualise their security levels.

It is demonstrated that ECC algorithm has a notably higher security level than RSA algorithm as ECC requires only a small key size to achieve the same level of security as RSA which requires a very large key size (Guruprakash & Koppu, 2020). Thus, we can see that the high encryption speed and low time complexity of ECC algorithm do not have a direct impact on its security level, see Figure 4.5.

In the following part, we will compare the processes taken for digital signing between EdDSA and ECDSA algorithms, as well as its consequences on their security.

As shown in Figure 4.6, it is clearly demonstrated that ECDSA requires more steps for single digital signing, and this is the reason leading to ECDSA consuming more time and resources for every process. To elaborate, instead of using a random number as in ECDSA, EdDSA algorithm directly uses a hash of the message. This ensures a different key to be generated for every message, as well as ensuring a collision-free system, thus resulting in improved security in EdDSA (Guruprakash & Koppu, 2021). The signature Rs is then computed by using the SHA algorithm, therefore it is of fixed length. Next, this signature Rs will be used together with the private key for generating a digital signature and it is then used to sign the message.

With reference to Wang, Yu, Zhang, Piao and Liu (2020), the well-known ECDSA weak randomness problem will directly affect the security of blockchain technology adversely. Since a random number is used in ECDSA algorithm for digital signatures, an unauthorised ECDSA signature may be generated by a user with a random number that is cryptographically insecure, what is worse is that the same random number may even be reused (Wang et al., 2020). This serious problem has a direct impact on the security of private keys that are important for protecting users’ finances and even the entire blockchain system (Wang et al., 2020). Likewise, if there is any computational error occurring in the middle of the ECDSA process, the private keys could also be found easily by the unauthorised users, which is very cryptographically insecure (Basha et al., 2021).

In short, not only the hashing of random numbers in the ECDSA algorithm causes it to be more time- and resource-consuming than the EdDSA algorithm, its mandate of using random numbers also makes ECDSA less efficient and more cryptographically breakable. Hence, it is obvious that the ECDSA algorithm should be further improved as its randomness indeed invites negative consequences towards its performance as well as security level. Not only that, despite the less computational processes taken by EdDSA, it is still considered more secure than ECDSA which requires extra steps in single digital signing.

Further suggestion to the enhancement of practicality of digital signatures would be keeping the latency and resources requirements low for digital signature algorithms to ensure the effectiveness of blockchain systems. This could probably be implemented by developing a more area-efficient and lower-cost ECC processor to make possible the deployment of multiple secure and advanced ECC algorithms in a single architecture without re-design, as proposed by Binh, Cuong, Tran, Pham, and Hoang (2022). It was found that certain combinations of ECC algorithms, if operated together, could result in a smaller delay of the processing time while the resources usage is kept constant (Binh et al., 2022). Perhaps removal of unnecessary processing elements would help in further reducing the required resources while maintaining the security level of blockchain applications.

Besides, another suggestion to the improvement of digital signatures would be reducing the power consumption of the digital signing process to the minimum, while simultaneously improving the accuracy of authentication in blockchain systems. This proposal is underpinned by Karthikeyan, Praghash, Raja, and Yuvaraj (2022), who discovered a novel mechanism integrating EdDSA algorithm with XOR function that is capable of offering better verification rate than other conventional algorithms including the EdDSA algorithm itself. Also, this proposed mechanism was found to have solely 100 to 120 milliseconds delay on average. Last but not least, the group affirmed that this novel mechanism manages to greatly lessen the energy consumption in the process of authenticating the nodes, which may invite significant benefits to the handheld devices due to their relatively smaller battery capacity.

On the security standpoint, although we discussed above on how ECDSA outperforms EdDSA of its same size as it is more resistant towards attacks, we have yet to talk about lattice-based fault attacks. Lattice based fault attacks exploit the physical features of deterministic signature schemes such as EdDSA and ECDSA by way of exploiting the algebraic properties that are present within the inner workings of a digital signature (Joye & Tunstall, 2012). This is accomplished by creating a lattice storing all the output of valid signatures and by injecting electrical faults in the system using electrical surges to create invalid signatures and then by comparing the valid signatures within the lattice with the invalid signatures the attacker is able to deduce the private key via the relationship that is present (Joye & Tunstall, 2012).

Specifically, a paper proposed that shortest vector problem (SVP) and closest vector problem (CVP) can be used in a lattice-based fault analysis method against some deterministic ECDSA and EdDSA algorithm to discover the signing key. Initially the ECDSA was thought to be resistant towards lattice-based fault attacks as they use Ephemeral random numbers (nonce), however they propose that the shortest problem within a lattice and the use of faulty signatures can be used to recover the key. The fault injection model describes how during the invocation of signature generation, the electrical signals can be tampered with to produce faulty signatures for the attacker. The attacker assumes that random fault of specific segment v of which w bits of v have been disturbed. After collecting a sufficient amount of tampered signatures, the attacker can construct a lattice that will have a short vector (discovered using the SVP) that is proportional to the private key (Cao et al., n.d.).

Additionally, one major issue of ECDSA is the need for unique nonce per signed message. This may pose a problem as when there is a reuse of the nonce, however as we come to know this issue has been mitigated by the deterministic generation of nonces (Jancar, Sedlacek, Svenda, & Sys, 2020). Additionally, the ECDSA is affected based on the vulnerable devices and libraries which it runs on, refer to Figure 4.7 (Jancar, Sedlacek, Svenda, & Sys, 2020). As we can see that the libraries SunEC, WolfSSL, libgcrypt, MatrixSSL, Crypto++ and Athena ID have leakage which affects the overall signature generation process as the ECDSA algorithm sits on the following libraries/devices. Taking into consideration, the scalar multiplication used on an elliptical curve of the Elliptic-Curve Diffie-Hellman (ECDH) the time in which this happens reveals the bit-length of the scalar, making it vulnerable to attacks, refer to Figure 4.8. By seeing how long a signature generation takes place and the nonce bit-length we can generate a heatmap that gives the attacker a clear idea of the range he/she needs to generate to carry out brute force attacks against the system. Hence, it is proposed that scalar randomisation methods be applied to ensure that some overhead is present to protect against power analysis attacks (Jancar, Sedlacek, Svenda, & Sys, 2020).

Encryptions

As a tradeoff of asymmetric encryption having significantly lower encryption and decryption speed as mentioned earlier in the “Performance and Usability Results” section above, it offers better security in encryption than that of symmetric encryption (Ahmad et al., 2015). To ensure the security of transferred files, asymmetric cryptography like RSA requires additional ciphers, more number of keys and larger key sizes to do the encryption and decryption in order to ensure the security of transferred files increase the complexity of ciphertexts after encryption (Ahmad et al., 2015). Therefore, this also explains the reason why asymmetric cryptography like RSA and ElGamal need more time and memory for the computation of keys (Abroshan, 2021), bringing to slower speed in performance than symmetric cryptography like AES.

Another factor of asymmetric cryptography having slow encryption and decryption speed is that asymmetric encryption involves difficult mathematical operations such as modular exponentiation used in RSA algorithms. However, due to these complex operations, asymmetric cryptography is more secure than symmetric cryptography as attackers are unable to solve these mathematical problems in feasible time. These complex mathematical problems play an important role in maintaining the high security of asymmetric encryption so that it cannot be cracked easily although they cause asymmetric encryption and decryption to become slower.

With that being said, it is clear that asymmetric cryptography such as RSA and ElGamal are not so practical in encrypting messages especially when the file size is large. Encrypting transactional data in blockchain by using asymmetric cryptography would definitely be slower than that of symmetric cryptography. This is also one of the dominant reasons why hybrid encryption is highly encouraged in the application of cryptography nowadays, including blockchain systems, to tackle the slow encryption problem caused by asymmetric cryptography and simultaneously minimise the security challenges invited by symmetric cryptography (Iavich, Gnatyuk, Jintcharadze, Polishchuk, & Odarchenko, 2018).

To elaborate, in hybrid encryption, symmetric key encryption (symmetric encryption) method is used to encrypt the data whereas public key encryption (asymmetric encryption) method is used to encrypt the symmetric session key. This maximises the data encryption speed and efficiency as well as improves the security of transactional data to its greatest extent. Not only that, the hybrid model comprising symmetric and asymmetric cryptosystems could result in lower power and memory consumption during data transactions in blockchain systems (Iavich et al., 2018).

In addition, another suggestion of improvement would be selecting the most appropriate hybrid encryption model for different blockchain platforms by comparing and evaluating their corresponding performance in terms of different metrics such as latency, throughput, security, memory and resources consumption, as well as time complexity. As supported by Abroshan (2021), study revealed that different combinations of symmetric and asymmetric encryption methods could result in different performance in which they provide different benefits to different platforms. Hence, the most suitable and pertinent hybrid model that performs particularly well on certain metrics should be given the main priority when it comes to choosing the most suitable hybrid model to implement based on the specific blockchain platform.

Appendix

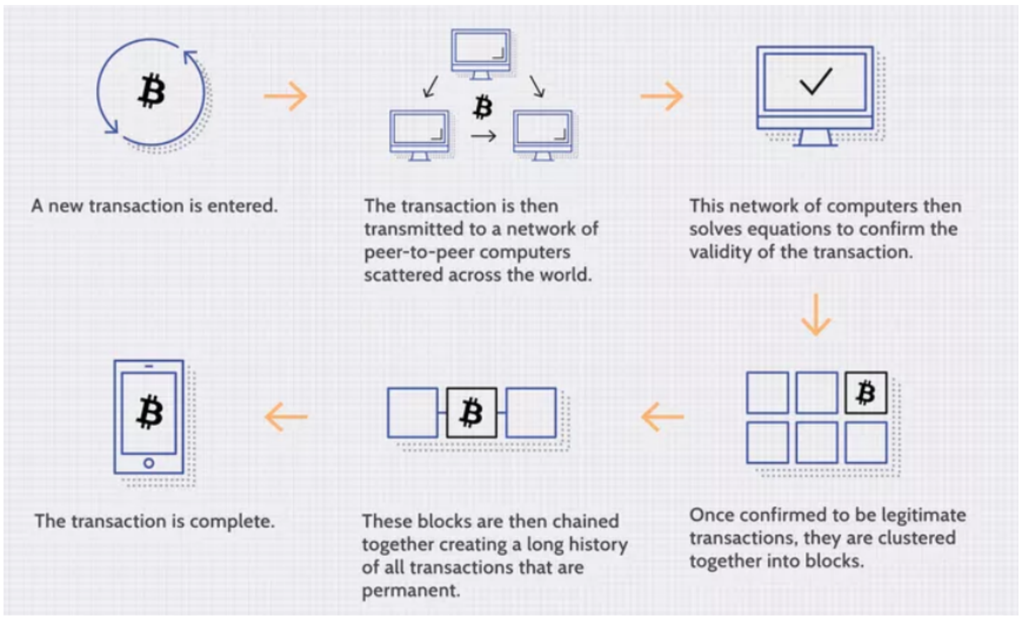

A new transaction is entered.
The transaction is then transmitted to a network of peer-to-peer computers scattered across the world.
This network of computers then solves equations to confirm the validity of the transaction.
Once confirmed to be legitimate transactions, they are clustered together into blocks.
These blocks are then chained together creating a long history of all transactions that are permanent.
The transaction is complete.

Figure 1.0 : Working of blockchain (Investopedia, n.d.).

Return to text

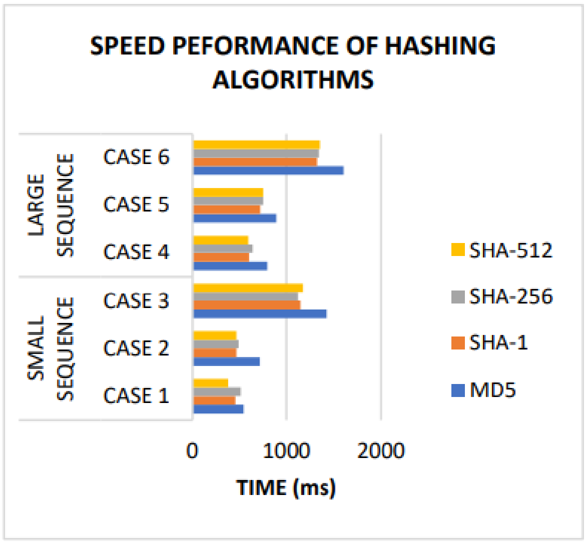
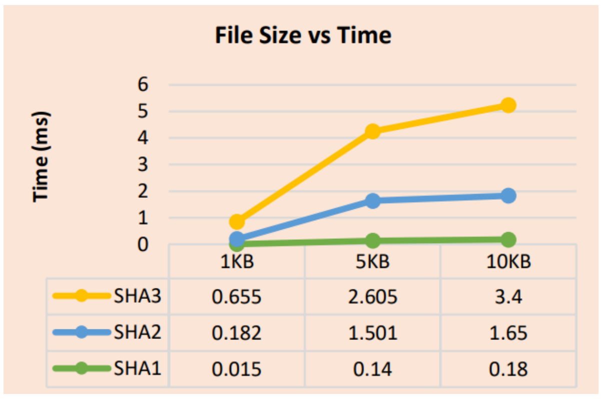


Figure 2.0 : Analysis of speed performance between different hash functions based on size of input data (Parmar & Jit, 2021).

Return to text



	1KB	5KB	10KB
SHA3	0.655	2.605	3.4
SHA2	0.182	1.501	1.65
SHA1	0.015	0.14	0.18

Figure 2.1: Analysis of execution time taken by different hash functions based on size of file (Parmar & Jit, 2021).

Return to text

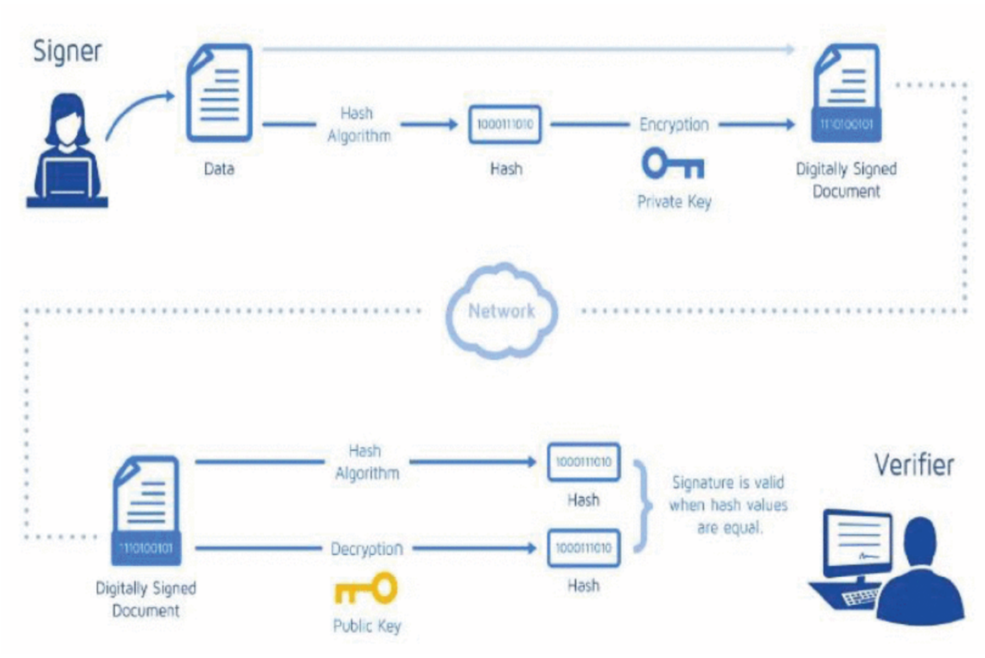


Figure 2.2: Process flow of Digital Signature (Basha et al., 2021).

Return to text

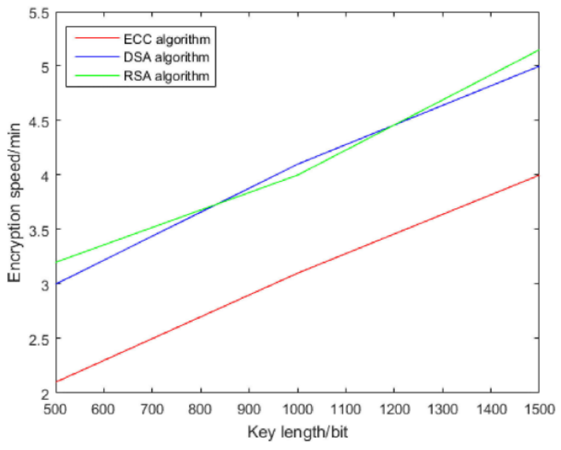


Figure 2.3: Speed of encryption based on key length among RSA, DSA and ECC algorithms (Guruprakash & Koppu, 2020).

Return to text

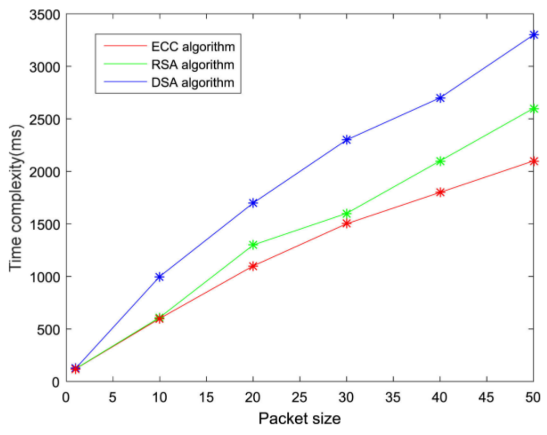


Figure 2.4: Time complexity based on packet size among RSA, DSA and ECC algorithms (Guruprakash & Koppu, 2020).

Return to text

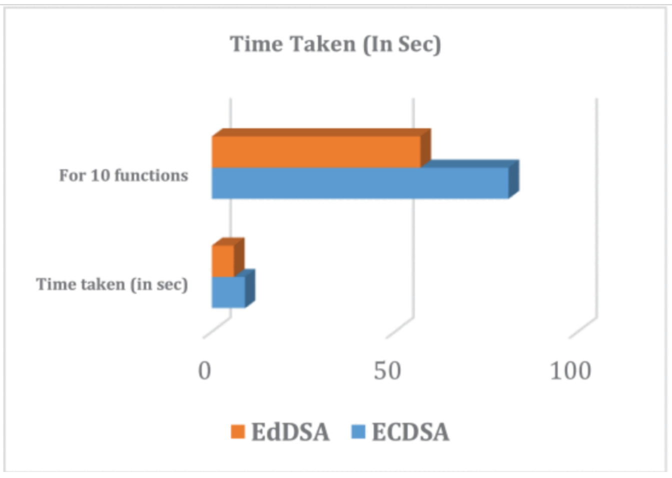


Figure 2.5: Time consumption taken for 1 and 10 hash functions respectively (Basha et al., 2021).

Return to text

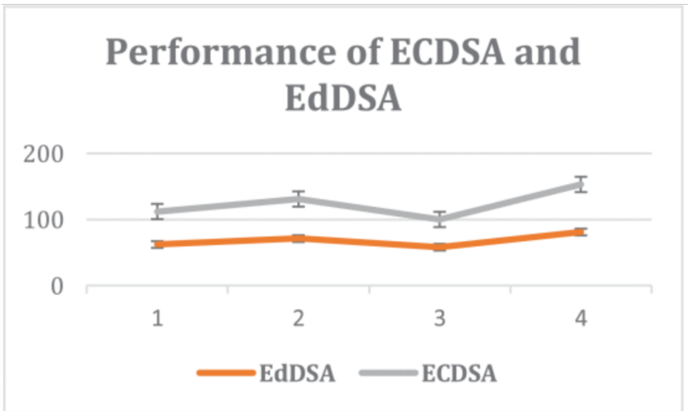


Figure 2.6: Evaluation of performance between ECDSA and EdDSA in terms of the number of computations required for single digital signing (Basha et al., 2021).

Return to text

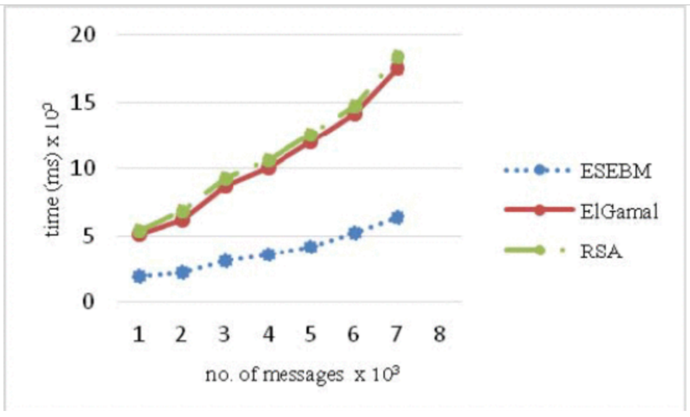
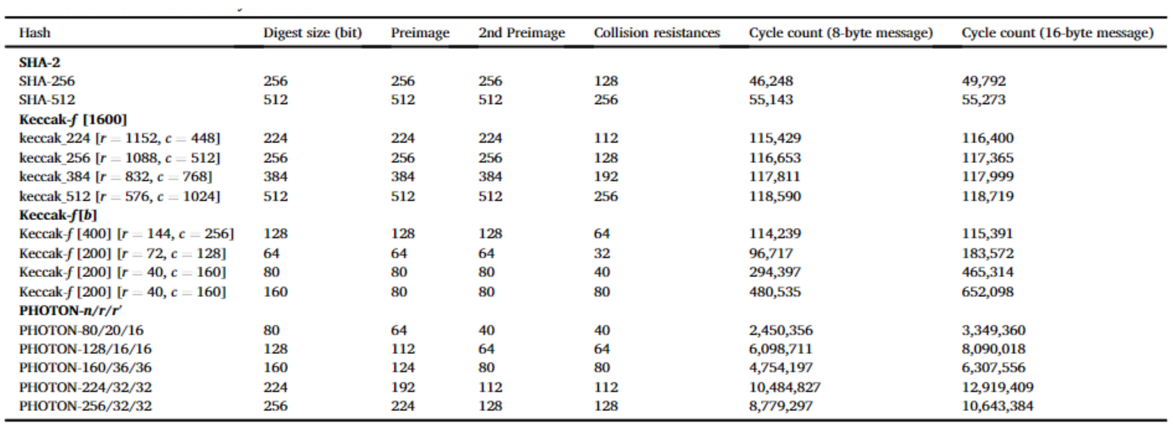


Figure 2.7: Decryption time used by ESEMB (symmetric encryption algorithm), RSA and ElGamal (asymmetric encryption algorithms) for 1024 bits key length from 1000 to 7000 messages (Ahmad, Alam, Rahman, & Tamura, 2015).

Return to text



Hash	Digest size (bit)	Preimage	2nd Preimage	Collision resistances	Cycle count (8-byte message)	Cycle count (16-byte message)
SHA-2						
SHA-256	256	256	256	128	46,248	49,792
SHA-512	512	512	512	256	55,143	55,273
Keccak-*f* [1600]						
keccak_224 [$r = 1152, c = 448$]	224	224	224	112	115,429	116,400
keccak_256 [$r = 1088, c = 512$]	256	256	256	128	116,653	117,365
keccak_384 [$r = 832, c = 768$]	384	384	384	192	117,811	117,999
keccak_512 [$r = 576, c = 1024$]	512	512	512	256	118,590	118,719
Keccak-*f*[*b*]						
Keccak-*f* [400] [$r = 144, c = 256$]	128	128	128	64	114,239	115,391
Keccak-*f* [200] [$r = 72, c = 128$]	64	64	64	32	96,717	183,572
Keccak-*f* [200] [$r = 40, c = 160$]	80	80	80	40	294,397	465,314
Keccak-*f* [200] [$r = 40, c = 160$]	160	80	80	80	480,535	652,098
PHOTON-*n*/*r*/*r*′						
PHOTON-80/20/16	80	64	40	40	2,450,356	3,349,360
PHOTON-128/16/16	128	112	64	64	6,098,711	8,090,018
PHOTON-160/36/36	160	124	80	80	4,754,197	6,307,556
PHOTON-224/32/32	224	192	112	112	10,484,827	12,919,409
PHOTON-256/32/32	256	224	128	128	8,779,297	10,643,384

Figure 3.0: Evaluation of hash functions in cycle count (Alfrhan, Moulahi, & Alabdulatif, 2021).

Return to text

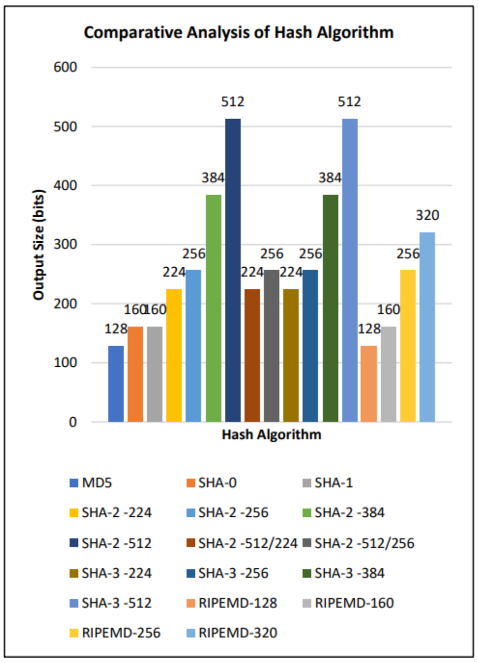
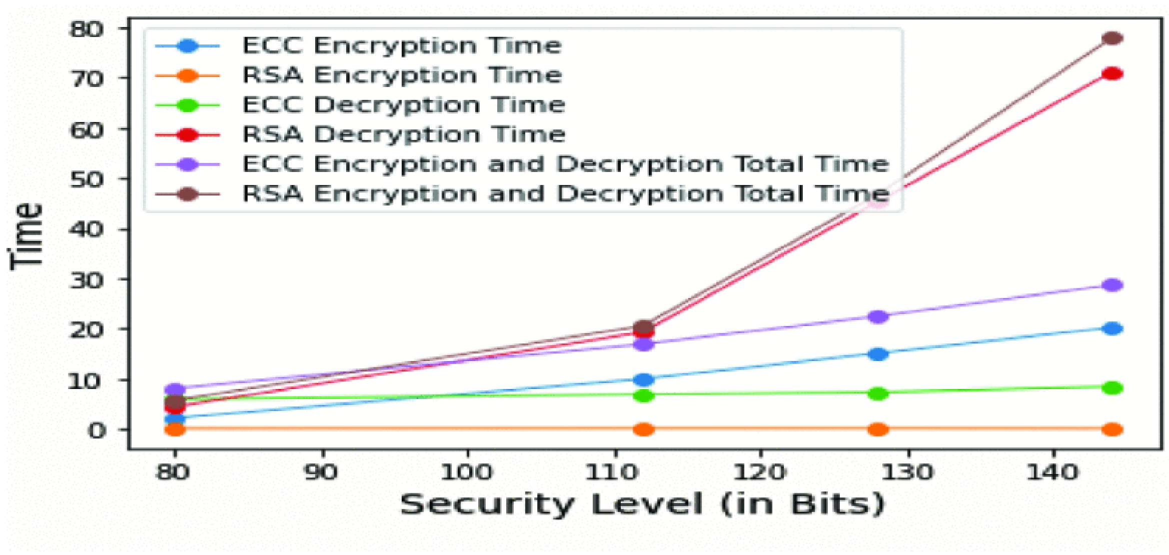


Figure 3.1: Comparative Analysis of Hash Algorithms (Parmar & Jit, 2021).

Return to text



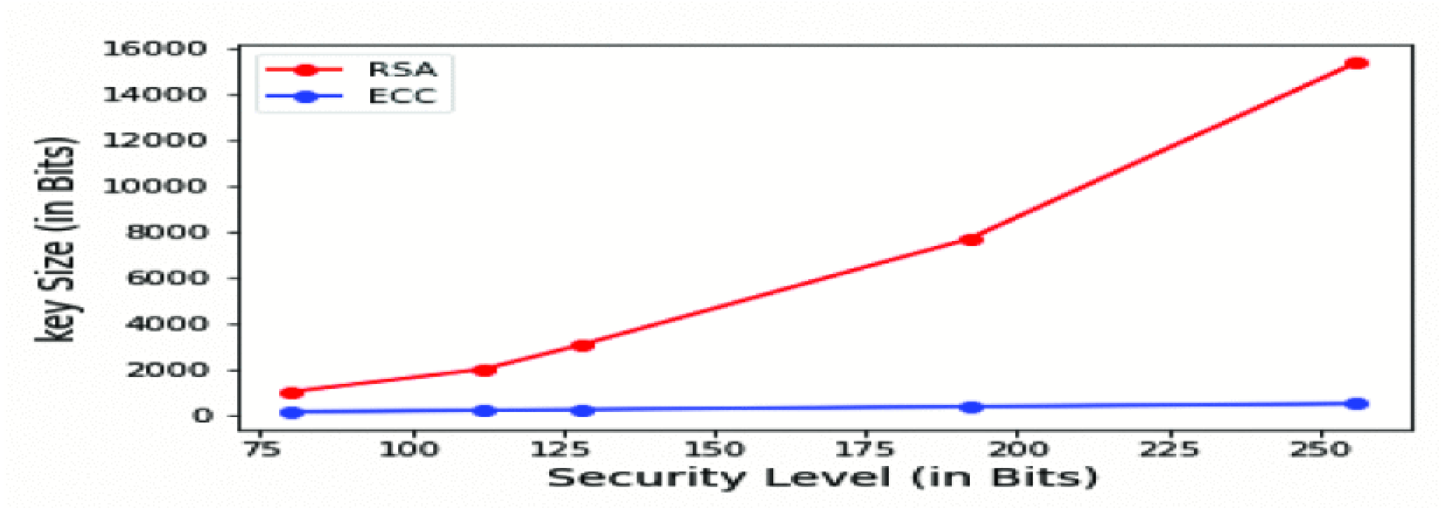


Figure 3.2: RSA and ECC (in Bits) with 64 Bit Encryption Decryption Time and Key Size against Security level (Yadav, 2021)

Return to text

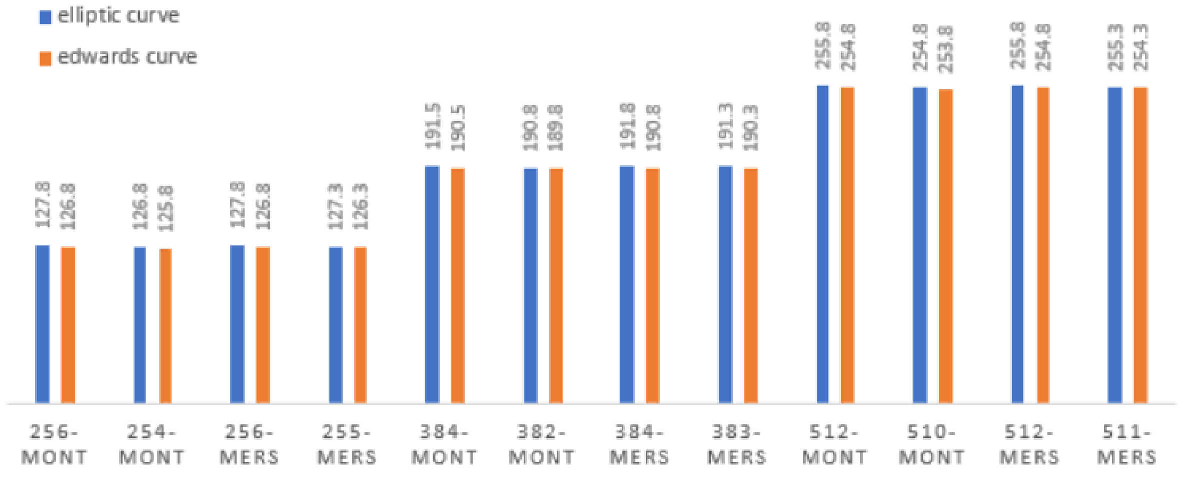


Figure 3.3: Security level comparison of Pollard's rho complexity (J & Koppu, 2022)

Return to text

```
Adversary $B$
    Input: $y, k$
    (1) $m_1 \leftarrow A(y,k)$,
    (2) If $|m_1| \neq \lambda$ or $Prefix_{|m_1|-c}(m_1) \neq 1^{|m_1|-c}$, then return $m_1$;
        else, the algorithm terminates.
```

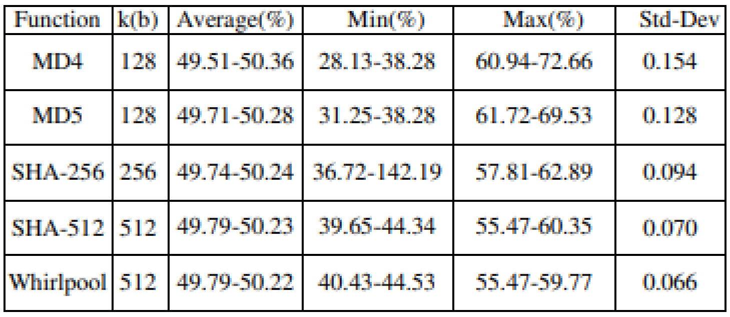
Figure 4.0: Las Vegas-type probability algorithm that either returns a hidden preimage or terminates (no value returned).

Return to text

$$\Pr_1 = \begin{cases} \frac{1}{2^{\tau}}, & if \quad \tau <|m_1|-c \quad and \quad Mid_{\tau+1,|m_1|-c}(m_1) = 1^{|m_1|-c-\tau} \\ 0, & if \quad \tau <|m_1|-c \quad and \quad Mid_{\tau+1,|m_1|-c}(m_1) \neq 1^{|m_1|-c-\tau} \\ \frac{1}{2^{|m_1|-c}}, & if \quad \tau \geq |m_1|-c \end{cases}$$

Figure 4.1: Probability function used to calculate the success rate of Adversary B attacking H.

Return to text



Function	k(b)	Average(%)	Min(%)	Max(%)	Std-Dev
MD4	128	49.51-50.36	28.13-38.28	60.94-72.66	0.154
MD5	128	49.71-50.28	31.25-38.28	61.72-69.53	0.128
SHA-256	256	49.74-50.24	36.72-142.19	57.81-62.89	0.094
SHA-512	512	49.79-50.23	39.65-44.34	55.47-60.35	0.070
Whirlpool	512	49.79-50.22	40.43-44.53	55.47-59.77	0.066

Figure 4.2: Sensitivity Test Results of Several Different Hash Functions

Return to text

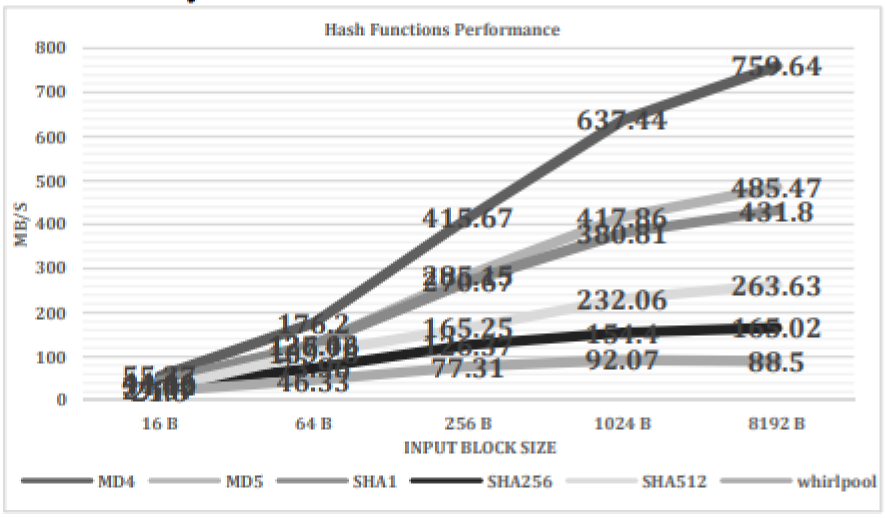
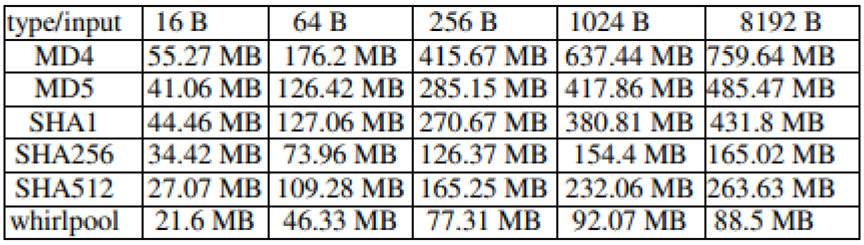


Figure 4.3 Hash performance graph comparing the MB/S throughput against the input block size for several different hash functions.

Return to text



type/input	16 B	64 B	256 B	1024 B	8192 B
MD4	55.27 MB	176.2 MB	415.67 MB	637.44 MB	759.64 MB
MD5	41.06 MB	126.42 MB	285.15 MB	417.86 MB	485.47 MB
SHA1	44.46 MB	127.06 MB	270.67 MB	380.81 MB	431.8 MB
SHA256	34.42 MB	73.96 MB	126.37 MB	154.4 MB	165.02 MB
SHA512	27.07 MB	109.28 MB	165.25 MB	232.06 MB	263.63 MB
whirlpool	21.6 MB	46.33 MB	77.31 MB	92.07 MB	88.5 MB

Figure 4.4 Hash performance tabulation comparing the MB/S throughput against the input block size for several different hash functions.

Return to text

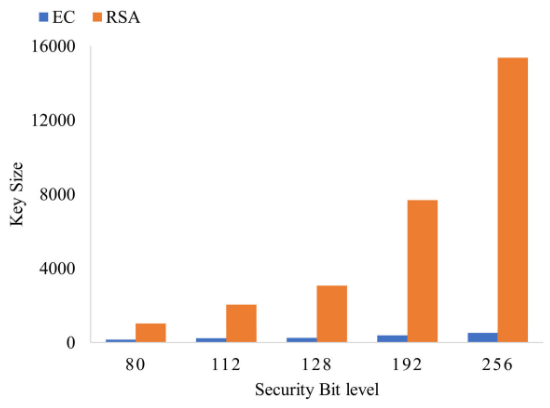


Figure 4.5: Key size based on security bit level among RSA and ECC algorithms (Guruprakash & Koppu, 2020).

Return to text

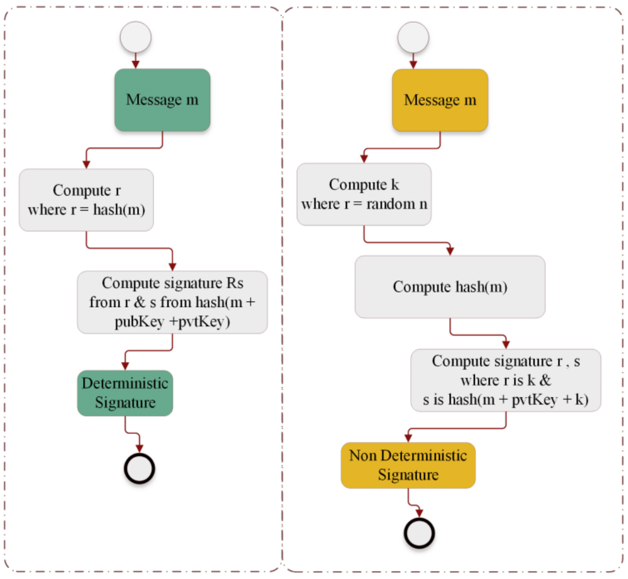
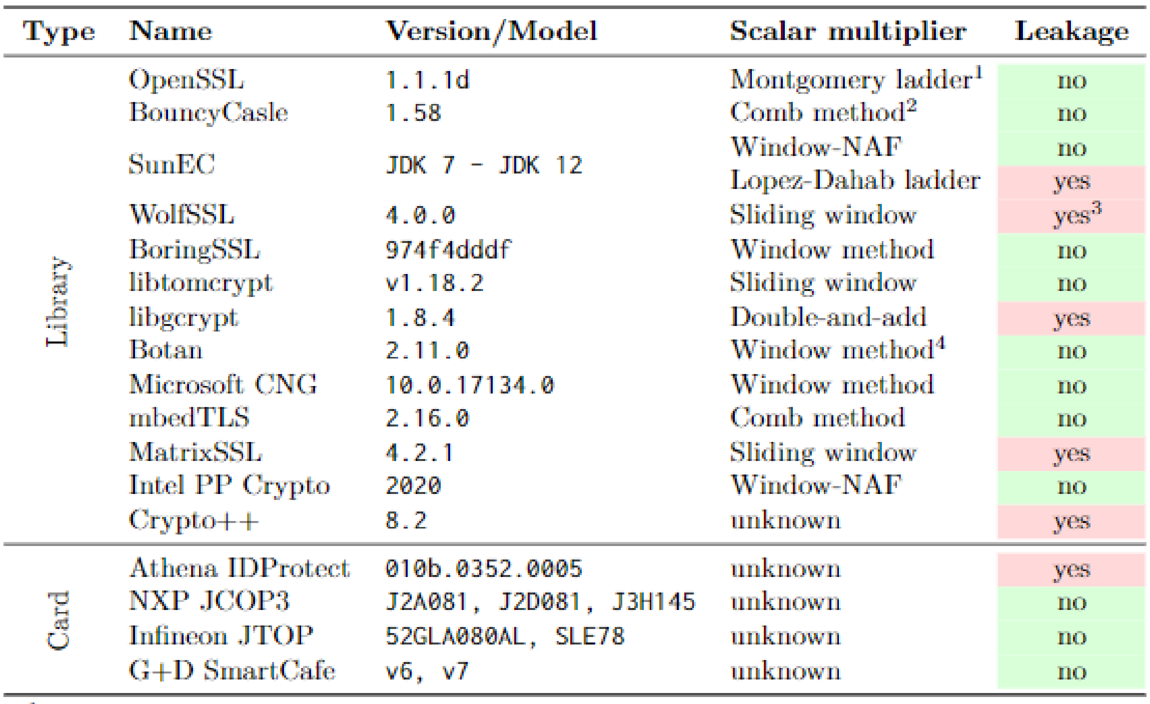


Figure 4.6: Process flow of EdDSA and ECDSA (Guruprakash & Koppu, 2021).

Return to text



Type	Name	Version/Model	Scalar multiplier	Leakage
Library	OpenSSL	1.1.1d	Montgomery ladder[1]	no
	BouncyCasle	1.58	Comb method[2]	no
	SunEC	JDK 7 - JDK 12	Window-NAF	no
			Lopez-Dahab ladder	yes
	WolfSSL	4.0.0	Sliding window	yes[3]
	BoringSSL	974f4dddf	Window method	no
	libtomcrypt	v1.18.2	Sliding window	no
	libgcrypt	1.8.4	Double-and-add	yes
	Botan	2.11.0	Window method[4]	no
	Microsoft CNG	10.0.17134.0	Window method	no
	mbedTLS	2.16.0	Comb method	no
	MatrixSSL	4.2.1	Sliding window	yes
	Intel PP Crypto	2020	Window-NAF	no
	Crypto++	8.2	unknown	yes
Card	Athena IDProtect	010b.0352.0005	unknown	yes
	NXP JCOP3	J2A081, J2D081, J3H145	unknown	no
	Infineon JTOP	52GLA080AL, SLE78	unknown	no
	G+D SmartCafe	v6, v7	unknown	no

Figure 4.7: Libraries and devices and their analysis in respects to leakage (Jancar, Sedlacek, Svenda, & Sys, 2020)

Return to text

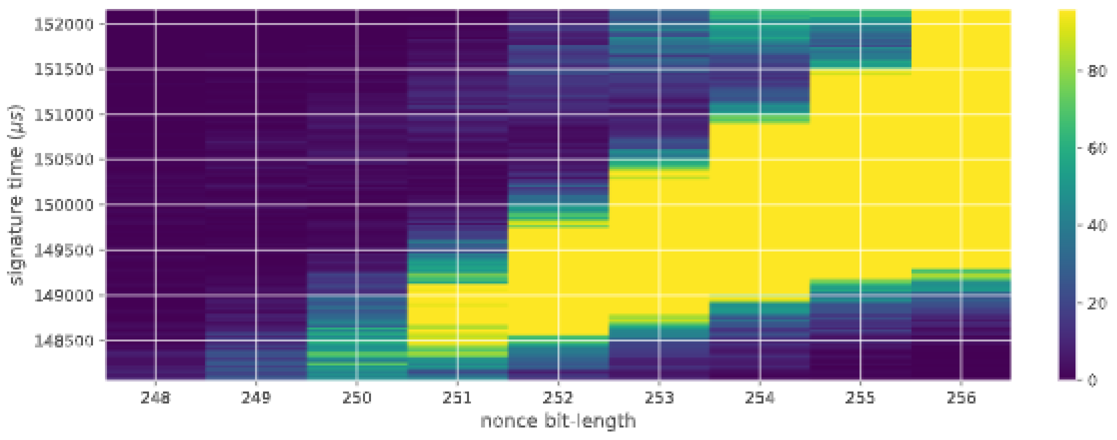


Figure 4.8: Heatmap of signing duration and bit-length of ECDSA nonces for 500 000 signatures using secp245r1 curve on the Athena IDProtect card (Jancar, Sedlacek, Svenda, & Sys, 2020)

Return to text

References

Abroshan, H. (2021). A Hybrid Encryption Solution to Improve Cloud Computing Security using Symmetric and Asymmetric Cryptography Algorithms. *International Journal of Advanced Computer Science and Applications*, *12*(6). https://doi.org/10.14569/IJACSA.2021.0120604

Ahmad, S., Alam, K. Md. R., Rahman, H., & Tamura, S. (2015). A comparison between symmetric and asymmetric key encryption algorithm based decryption mixnets. *2015 International Conference on Networking Systems and Security (NSysS)*, 1–5. Dhaka, Bangladesh: IEEE. https://doi.org/10.1109/NSysS.2015.7043532

Alfrhan, A., Moulahi, T., & Alabdulatif, A. (2021). Comparative study on hash functions for lightweight blockchain in Internet of Things (IoT). *Blockchain: Research and Applications*, *2*(4), 100036. https://doi.org/10.1016/j.bcra.2021.100036

Amy, M., Di Matteo, O., Gheorghiu, V., Mosca, M., Parent, A., & Schanck, J. (2017). Estimating the Cost of Generic Quantum Pre-image Attacks on SHA-2 and SHA-3. In R. Avanzi & H. Heys (Eds.), *Selected Areas in Cryptography – SAC 2016* (pp. 317–337). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-69453-5_18

Basha, S. J., Veesam, V. S., Ammannamma, T., Navudu, S., & Subrahmanyam, M. V. V. S. (2021). Security Enhancement of Digital Signatures for Blockchain using EdDSA Algorithm. *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, 274–278. Tirunelveli, India: IEEE. https://doi.org/10.1109/ICICV50876.2021.9388411

Benetton, M., Compiani, G., & Morse, A. (2021). When Cryptomining Comes to Town: High Electricity-Use Spillovers to the Local Economy. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3779720

Cao, W., Shi, H., Chen, H., Chen, J., Fan, L., & Wu, W. (n.d.). *Lattice-based Fault Attacks on Deterministic Signature Schemes of ECDSA and EdDSA*. Retrieved from https://eprint.iacr.org/undefined/undefined

Chandran, N. R., & Manuel, E. M. (2016). Performance Analysis of Modified SHA-3. *Procedia Technology*, *24*, 904–910. https://doi.org/10.1016/j.protcy.2016.05.168

Dilhara, B. A. S. (n.d.). *A Review on Application of Hash Functions and Digital signatures in the Blockchain Industry*.

Fang, W., Chen, W., Zhang, W., Pei, J., Gao, W., & Wang, G. (2020). Digital signature scheme for information non-repudiation in blockchain: A state of the art review. *EURASIP Journal on Wireless Communications and Networking*, *2020*(1), 56. https://doi.org/10.1186/s13638-020-01665-w

Ghazal, T. M., Hasan, M. K., Abdullah, S. N. H. S., Bakar, K. A. A., & Al Hamadi, H. (2022). Private blockchain-based encryption framework using computational intelligence approach. *Egyptian Informatics Journal*, *23*(4), 69–75. https://doi.org/10.1016/j.eij.2022.06.007

Guruprakash, J., & Koppu, S. (2020). EC-ElGamal and Genetic Algorithm-Based Enhancement for Lightweight Scalable Blockchain in IoT Domain. *IEEE Access*, *8*, 141269–141281. https://doi.org/10.1109/ACCESS.2020.3013282

Han, R., Foutris, N., & Kotselidis, C. (2019). Demystifying Crypto-Mining: Analysis and Optimizations of Memory-Hard PoW Algorithms. *2019 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 22–33. Madison, WI, USA: IEEE. https://doi.org/10.1109/ISPASS.2019.00011

Iavich, M., Gnatyuk, S., Jintcharadze, E., Polishchuk, Y., & Odarchenko, R. (2018). Hybrid Encryption Model of AES and ElGamal Cryptosystems for Flight Control Systems. *2018 IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC)*, 229–233. Kiev, Ukraine: IEEE. https://doi.org/10.1109/MSNMC.2018.8576289

IBM. (2022, November 9). IBM Unveils 400 Qubit-Plus Quantum Processor and Next-Generation IBM Quantum System Two. Retrieved 2 April 2023, from IBM Newsroom website: https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two

Investopedia. (n.d.). Blockchain Facts: What Is It, How It Works, and How It Can Be Used. Retrieved 5 April 2023, from Investopedia website: https://www.investopedia.com/terms/b/blockchain.asp

J, G., & Koppu, S. (2022). An empirical study to demonstrate that EdDSA can be used as a performance improvement alternative to ECDSA in Blockchain and IoT. *Informatica*, *46*(2). Retrieved from https://www.informatica.si/index.php/informatica/article/view/3807

Jancar, J., Sedlacek, V., Svenda, P., & Sys, M. (2020). Minerva: The curse of ECDSA nonces : Systematic analysis of lattice attacks on noisy leakage of bit-length of ECDSA nonces. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 281–308. https://doi.org/10.13154/tches.v2020.i4.281-308

Joye, M., & Tunstall, M. (Eds.). (2012). *Fault Analysis in Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29656-7

Khanal, Y. P., Alsadoon, A., Shahzad, K., Al-Khalil, A. B., Prasad, P. W. C., Rehman, S. U., & Islam, R. (2022). Utilizing Blockchain for IoT Privacy through Enhanced ECIES with Secure Hash Function. *Future Internet*, *14*(3), 77. https://doi.org/10.3390/fi14030077

Kieu-Do-Nguyen, B., Pham-Quoc, C., Tran, N.-T., Pham, C.-K., & Hoang, T.-T. (2022). Low-Cost Area-Efficient FPGA-Based Multi-Functional ECDSA/EdDSA. *Cryptography*, *6*(2), 25. https://doi.org/10.3390/cryptography6020025

Konashevych, O., & Poblet, M. (n.d.). *Is Blockchain Hashing an Effective Method for Electronic Governance?*

Maliberan, E. V. (2022). Modified SHA1: A Hashing Solution to Secure Web Applications through Login Authentication. *International Journal of Communication Networks and Information Security (IJCNIS)*, *11*(1). https://doi.org/10.17762/ijcnis.v11i1.3726

Monrat, A. A., Schelen, O., & Andersson, K. (2019). A Survey of Blockchain From the Perspectives of Applications, Challenges, and Opportunities. *IEEE Access*, *7*, 117134–117151. https://doi.org/10.1109/ACCESS.2019.2936094

Parmar, M., & Jit, H. (2021). Comparative Analysis of Secured Hash Algorithms for Blockchain Technology and Internet of Things. *International Journal of Advanced Computer Science and Applications*, *12*(3). https://doi.org/10.14569/IJACSA.2021.0120335

Pillai, B., Hou, Z., Biswas, K., & Bui, V. (2022). *Blockchain Interoperability: Performance and Security Trade-offs*.

Rahulamathavan, Y., Phan, R. C.-W., Rajarajan, M., Misra, S., & Kondoz, A. (2017). Privacy-preserving blockchain based IoT ecosystem using attribute-based encryption. *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 1–6. Bhubaneswar: IEEE. https://doi.org/10.1109/ANTS.2017.8384164

Rogaway, P., & Shrimpton, T. (2004). Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In B. Roy & W.

Meier (Eds.), *Fast Software Encryption* (pp. 371–388). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-25937-4_24

Rubayya, R. S., & Resmi, R. (2014). Memory optimization of HMAC/SHA-2 encryption. *2014 First International Conference on Computational Systems and Communications (ICCSC)*, 282–287. Trivandrum, India: IEEE. https://doi.org/10.1109/COMPSC.2014.7032663

Sklavos, N. (2010). On the Hardware Implementation Cost of Crypto-Processors Architectures. *Information Security Journal: A Global Perspective*, *19*(2), 53–60. https://doi.org/10.1080/19393551003649016

Subramanya, S. R., & Yi, B. K. (2006). Digital signatures. *IEEE Potentials*, *25*(2), 5–8. https://doi.org/10.1109/MP.2006.1649003

Wang, M., Duan, M., & Zhu, J. (2018). Research on the Security Criteria of Hash Functions in the Blockchain. *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts*, 47–55. Incheon Republic of Korea: ACM. https://doi.org/10.1145/3205230.3205238

Wang, Z., Yu, H., Zhang, Z., Piao, J., & Liu, J. (2020). ECDSA weak randomness in Bitcoin. *Future Generation Computer Systems*, *102*, 507–513. https://doi.org/10.1016/j.future.2019.08.034

Wu, J., & Tran, N. (2018). Application of Blockchain Technology in Sustainable Energy Systems: An Overview. *Sustainability*, *10*(9), 3067. https://doi.org/10.3390/su10093067

Yadav, A. K. (2021). Significance of Elliptic Curve Cryptography in Blockchain IoT with Comparative Analysis of RSA Algorithm. *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 256–262. https://doi.org/10.1109/ICCCIS51004.2021.9397166