LEX LOCALIS-JOURNAL OF LOCAL SELF-GOVERNMENT
ISSN:1581-5374 E-ISSN:1855-363X
VOL. 23, NO. 11(2025)

A SECURE AND SCALABLE DIGITAL SIGNATURE PROTOCOL FOR IOT APPLICATIONS

U. Siddharth Nambi[1,2], R. Kishore[3], V. Manickamuthu[4]

[1]Assistant Professor, Department of ECE,
J.N.N Institute of Engineering, Tamilnadu, India
[2]Research Scholar, Department of ECE,
Sri Sivasubramaniya Nadar college of Engineering, Tamilnadu, India
[3]Associate Professor, Department of ECE,
Sri Sivasubramaniya Nadar college of Engineering, Tamilnadu, India
[4]PG Student, Department of ECE,
Sri Sivasubramaniya Nadar college of Engineering, Tamilnadu, India

Corresponding email: siddharthnambi3@gmail.com[1]

***Abstract*—**Despite the enormous potential of Internet of Things (IoT) in the diverse spheres, the communication infrastructure is flawed from the security standpoint and inclines to loss of privacy for end-users. Many security algorithms have been explored for providing secure communication, but battery-operated smart devices can support only the lightweight operations. This work aims to introduce a lightweight security protocol for resource-constrained settings, leveraging complex numbers derived from small primes in a defined range to compute signature parameters, enhanced by SHA-1 for improved resilience. The probability to share a key between devices could be increased by suitably selecting the prime number range. The connectivity and resilience against node capture are evaluated to know the performance of the algorithm.

***Index Terms*—**IoT, Security, Lightweight, DSA, Digital Signature, Resilience, Connectivity, SHA, Network, communication.

I. Introduction

Internet is a massive network to facilitate the exchange of information in all communication areas. Due to its wide range of applications, the Internet of Things (IoT) is gaining a huge response nowadays. It comprises of a large number of smart devices to share the sensed data over the internet and to cache at cloud repositories. It has become a necessity for future communication. For example, the information about home automation, transportation domain, automobile domain, smart grid, and smart meters are all based on IoT. The different phases involved in IoT system design are data collection, acquisition, perception, storage, intelligent processing, data transmission, and delivery. The foremost phase is to collect data from devices or things. A device may be a static or a dynamic body [1]. Based on the attributes of the device, different types of sensors are used to collect data. Then the collected data should be stored on remote cloud storage systems. In general, IoT devices are resource-constrained and equipped with less memory and processing capabilities. In addition to that, they also have an Internet Protocol address (IP address). It is the responsibility of the cloud to store the data forwarded by these devices. Then, the data in the cloud are analyzed by the IoT and accordingly it provides the required services in a real-time environment. In all the phases of IoT system design, the data transmission occurs and the processed data should be delivered to the devices on time without error. The collected data from the IoT devices are sent to the cloud where it can be accessed by the concerned recipients and also the data should possess its originality.

Irrespective of the nodes in the network, the operations should be carried out to make the data secure and to make sure the effective reception in the receiving end. The hacking, interception in data, spoofing, eavesdropping or human error are some of the malicious activities which may happen during the exchange of data in the IoT applications. To overcome these malicious behaviors, the confidentiality, authentication, integrity, and freshness of the data are the needs to ensure security in the IoT. Data confidentiality is the omission of data leaks to neighboring networks. Data authentication means the verification of the corresponding sender and receiver. Data integrity deals with the non-altered transmission of data. Data freshness ensures that the data is recent while allowing for delay estimation [2].

Key challenges in security involve striking an optimal balance between reducing resource use and enhancing protection levels. Security mechanisms on devices rely on their hardware capabilities and limitations, so resource constraints must be addressed. IoT networks lack fixed topology, enabling attackers to conduct passive eavesdropping, active interference, secret leaks, message tampering, node impersonation, and more. The digital signatures are used to achieve authentication where the private key is used for signing and a public key is used for verification. It also uses hash algorithms to make the data more secure and to produce signature parameters. Digital signatures of asymmetric cryptography are a good option to ensure the authenticity and originality of the communicating devices. Digital Signature Algorithm (DSA) has been used for interchanging the data, distribution of software and storing the information. It consists of the signature generation and verification algorithms. The digital signature is mainly applicable in the following two scenarios. 1) Direct digital signature, where the data are important and private to the end-user, like business transactions. Here, user A signs the data and transmits to user B which verifies the signature to access the data. 2) Arbitrated digital signature, in which the signed data are sent to an arbiter first to check the integrity and authenticity of the data and then it is forwarded to the intended receiver. Both of the schemes use the s parameter of signature (r, s) in modified form in the verification process.

A. Motivation

The most prominent threats affecting the entire developing IoT systems arise out of security issues present in the technologies used in the IoT to relay the information from one device to another device. The privacy of users and their data protection has been identified as one of the important challenges which need to be addressed in the IoT.

B. Objective

To provide secure communication, several signature-based schemes were explored in literature but smart devices need more lightweight operations by ensuring desired security strengths. On this note, the objective of the proposed work is to identify the security and privacy challenges in IoT and develop a lightweight security protocol for constrained environments.

The rest of the paper follows this structure. Section II reviews prior research. Section III introduces Digital Signature Algorithms. Section IV details the new approach, including simulation and experimental outcomes. Section V wraps up with conclusions and ideas for future extensions.

II. RELATED WORKS

The IoT network or device is not physically protected and thus the information exchanged can be forged easily. To protect the data from the attackers, many algorithms are introduced and few of them are discussed in this section. The energy-efficient algorithm with low computational complexities is preferred in the IoT because of the resource-constrained devices.

Kamal and Tariq [3] have proposed a lightweight protocol for secure communication and also to detect the presence of an adversarial node in the network. The link fingerprints were generated using the Received Signal Strength Indicator (RSSI) which is unique for every link. An adversarial node was identified by correlating the link fingerprints generated by the neighboring IoT nodes. The correlation coefficient was computed at the server in which the RSSI was used to find the link fingerprint and the correlation percent gave the presence of adversarial node. The simulations on adversarial node detection, data provenance and time-complexity were performed. The authors inferred that, the situation with correlation coefficient greater than 90% depicts the absence of an adversarial node. To produce a lengthy link fingerprint, high energy was consumed. So optimized techniques may be used to overcome this energy consumption.

Wazid et al. [4] proposed a lightweight three-factor authentication scheme for a remote user in the Hierarchical IoT Networks (HIoTNs), called the User Authenticated Key Management Protocol (UAKMP) for the case of Dolev Yao (DY) threat model. The three factors were user smart card, password, and personal biometrics. The simulations on Real or Random (ROR) models were done and the resilience of UAKMP

against various attacks was analyzed. The UAKMP commonly runs in the HIoTNs. Instead, it could be run in all the nodes for better secrecy. It has a high computational cost as compared to traditional methods.

In [5], Mughal et al. proposed a Shortened Complex Digital Signature Algorithm (SCDSA) that uses shortened complex numbers for signature and verification operations for the traffic analysis attack. The author also proposed a Multi option Parameter Selection (MPS-SCDSA) which provided options to the user in the selection of signature/verification expressions pair. The simulation results show that the algorithm gave less computational time of 922.934 (ms), communication overhead reduction of 60% per message and improved security to provide Data Leakage Prevention (DLP). Their future work includes intrusion detection capabilities to measure timely attack detection and prevention.

Chen et al. [6] presented a lightweight and anonymous authentication protocol using a certificateless unidirectional proxy re-signature scheme to overcome the Computational Diffie Hellman (CDH) problem for mobile payment. The author adopted batch verification to mitigate the overhead for millions of users and also addressed the scalability issue. The random oracle model was used to enhance the transaction efficiency but it increases the computational complexity.

Pu and Lim [7] proposed a light-weight countermeasure called SCAD for the Distributed Denial of Service (DDoS) attack in which a single checkpoint node was deployed to detect the malicious nodes. The proposed scheme was combined with a hop-by-hop retransmission and a timeout technique to recover unexpected packet losses. The simulation results depicted the improvement in detection rate and packet delivery ratio as well as reduction in the energy consumption and false detection rate. This method will not hold good for dynamically changing routes in case of traffic inconvenience.

In [8], Li et al. proposed a policy-based trustworthy algorithm named RealAlert to address the security issues in IoT and to detect the malicious nodes. It is a comprehensive scheme with four components - data collection, policy management, malicious node detection, and trust management. Based on the history and context of the data, the trustworthiness was calculated and the data are collected using the policy rules to identify the malicious nodes. It achieved 80% of precision and recall scores eventhough there were 40% of malicious nodes that were conducting bad mouth attacks but requires more memory to run the policies.

Giuliano et al. proposed an algorithm based on a secure key renewal for the secure access of unidirectional and bidirectional IP and non-IP devices, considering a local clock time and a time interval of key validity which manages and renews the keys efficiently [9]. The problem of coexistence between ALOHA and CSMA terminals transmitting in the same area has been analyzed by simulations. For the specific performance target, it discusses the maximum number of ALOHA and CSMA based terminals that can be installed in a particular area. Anyhow for all the scenarios, the total probability of the number of ALOHA based terminals and the number of CSMA based terminals will be one.

Antonio Pintoa and Ricardo Costa [10] proposed a modified hash-chain authentication mechanism that can authenticate each interaction of the devices with a REST web-service using One Time Passwords (OTP) within a predefined time window while using open wireless networks. On the client-side, the login procedure using the SHA-256 took 19.3ms to conclude and the SHA-512 took 19.1ms to conclude the login procedure. It enabled anti-replay protection and prevented both man-in-the-middle and denial of service attacks. Even though SHA-512 took lesser time to compute hash code, more bandwidth was required for execution compared to SHA-256. Thus, there will be a tradeoff between the time and the bandwidth while using the SHA algorithms.

The several schemes in the literature discuss on providing a secure communication in Wireless Sensor Networks (WSN) using secure key management, policies, and signatures. It also discusses the various

constraints such as memory and energy usage, computational complexity and cost. In this paper, the light weight digital signature scheme is proposed for IoT in which the small complex numbers are used for computing the signature and it is implemented in the raspberry pi to get the real time results. The performance is analyzed in terms of connectivity and resilience. From the analysis, it can be understood that the proposed scheme gives a better resilience using lightweight signatures thereby overcomes IoT constraints, such as finite resource, limited memory, and less processing capabilities.

III. DIGITAL SIGNATURE ALGORITHMS

In the real-time application, the end-user needs assurance that the data are from the appropriate sender and the receiver should not be able to repudiate the origin of the data. This plays a vital role in business applications since the likelihood of the dispute over exchanged data is very high. Therefore, to ensure the authentication of the data, digital signatures are used and it employs asymmetric cryptography. Initially, the sender signs the data before transmitting it to the recipient. It also uses the hash of the data to detect the modifications by malicious nodes [11]. The signature and hash of the data are bind together and sent to a recipient. This binding is independently verified by a receiver or any other third party also. If the signature is verified by the receiver, the receiver can access the data sent by the sender otherwise the receiver cannot access it.

A. Digital Signature Standard

The Digital Signature Standard (DSS) makes use of the Secure Hash Algorithm (SHA) and presents a digital signature technique, called the DSA. It is based on the mathematical concept of modular exponentiation and the discrete logarithm problem.

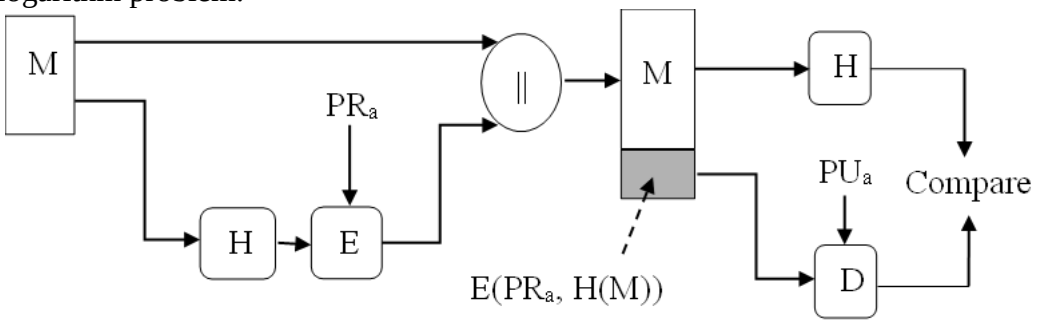


Fig. 1. RSA approach [12].

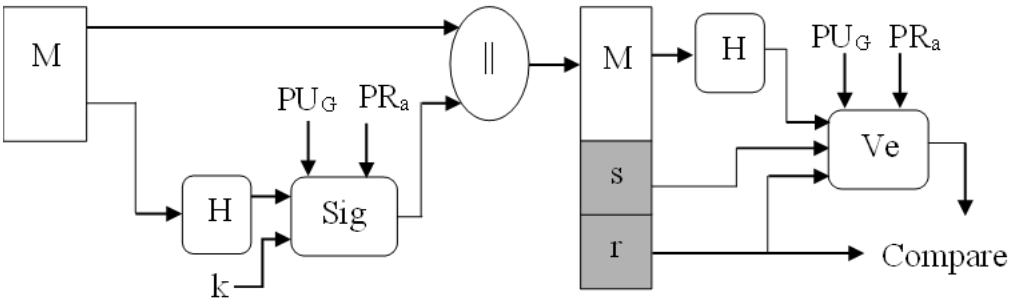


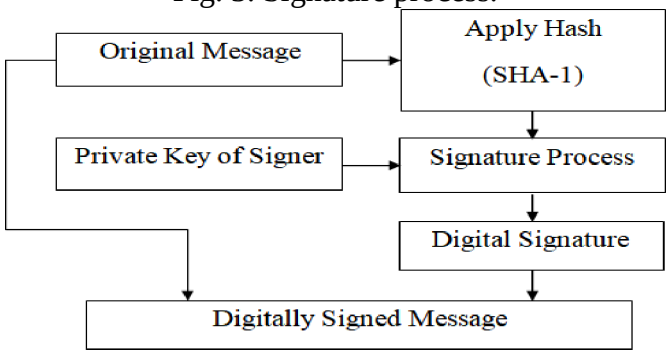
Fig. 2. DSS approach [12].

1. DSS Approach: The DSS approach uses an algorithm that presents only the digital signature operation. Unlike Rivest Shamir Adleman (RSA), it cannot be used for encryption. Fig. 1 and Fig. 2 depict the RSA approach and DSS approach respectively for generating digital signatures. In the RSA approach, the hash function takes the data to be signed as an input and produces a hash value as an output. To get the signature, the sender's private key is used to encrypt the output and then the data are transmitted along with the signature to the recipient. The recipient receives the signature and uses the public key of the sender to decrypt the data. It generates the hash value using the signature parameters. If the generated hash value matches with the

decrypted signature, the signature is taken as valid otherwise it is invalid [12]. In the DSS approach, the signature operation uses the hash value as an input along with a unique random number *k* produced for the signature. The sender's private key (PR_a) and the set of parameters known to a particular group of communicating devices are also used by the signature operation. By using those parameters, the global public key (*PUG*) can also be found. The output of the signature operation consists of two components, labeled *r*, and *s*. The verification function takes the signature as input and the hash value of the received data is also generated at the receiver side. It also uses the global public key *(PUG)* and the public key of the sender (PU_a). If the output of the verification operation equals one of the received signature component *r*, the signature is valid and the receiver can access the message.

2) *Digital Signature Algorithm:* DSA is a variant of the Schnorr and ElGamal signature schemes. It comprises of signature generation and verification algorithm and the process is depicted in Fig. 3 and Fig. 4 respectively. The steps to achieve the signature and verification in DSA are key generation, signing, verifying a signature, correctness of the algorithm. The two steps involved in a key generation are parameter selection and per-user keys generation. Each device with DSA has a public-private key pair and the key pairs of signing and verifying processes differ from the encryption and decryption. The message to be transmitted is fed to the hash function and it produces the hash code as output. The signature algorithm uses a private key and the hash code to generate the digital signature. The message is attached with the signature and sent to a receiver. The receiver or verifier receives the digital signature and uses it along with the verification key into the verification algorithm. To get the hash code, the same hash algorithm is applied to the received message. The output of the verification algorithm is compared with the received digital signature to verify the signature in the receiver device. Based on the compared output, the receiver concludes whether the digital signature is valid or not. The digital signature is created by the signer's private key so no one else can hold this key; thus, in the future, the signer cannot repudiate signing the message. The usage of modular exponentiation for signing a huge message is time-consuming and computationally expensive. Thus, the hash algorithm is used for signing in DSA. The hash of the message is a relatively small digest of the message, hence signing a hash is more efficient than signing the entire message. The global public key components are considered according to the algorithm in Fig. 5 and in that M represents the message, H(M) defines the hash of message M to be signed using SHA-1. The received forms of r, s, and M are r', s', and M' respectively.

Fig. 3. Signature process.



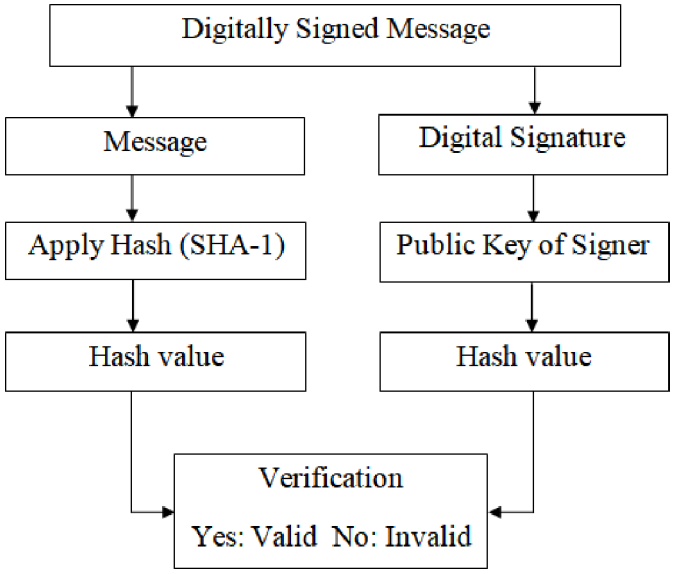


Fig. 4. Verification process.

3) DSA Signature Generation: The r and s are the signature parameters of message M that are computed according to the signing equation in Fig.5. The value of SHA-1(M) is a 160-bit string output. For computing the second parameter of the signature s, the SHA-1 output must be modified into an integer. The condition r = 0 or s = 0 may also be checked. If either r = 0 or s = 0, a new k should be produced and the signature need to be regenerated where k is a randomly generated integer. Finally, the message is transmitted along with the signature to the recipient.

4) DSA Signature Verification: Before verification of the signature, the parameters p, q, and g are made available along with the sender's public key to the verifier. The received forms of r, s, and M are r', s', and M' respectively and y be the public key of the signer. First, the conditions $0 < r' < q$ and $0 < s' < q$ are checked and if any one of the conditions is not satisfied, the signature will be rejected. If the conditions are satisfied, the verifier will compute the parameters for a verification process that is specified in Fig.5. The verification of the signature happens only if $V = r'$ and the receiver can also have confidentiality about the received message. If V is not equal to r', the message may be signed by the signer incorrectly or it may be modified. Thus, the message will be considered as an invalid message.

B. Importance of Digital Signature

To accomplish data security, the DSA with public key cryptography is the rewarding and significant strategy among all essential techniques of cryptography. The data authentication, integrity and non-repudiation are the key features of digital signature [13].

1) Message Authentication: In the case of single information like an alarm signal, the authentication function ensures the intended recipient that the message is from the origin. In the event of continuous interactions like connecting the source and the terminal, the authentication function provides two features. First, it ensures that the communicating nodes are authentic during connection establishment and then it also ensures that there is no interference of any third party which may imitate as one of the legitimate nodes for illegal transmission or reception. In DSA, when the verifier checks the validity of the digital signature using the sender's public key, the receiver is guaranteed that the signature has been created only by the sender who possesses the corresponding private key and no one else [14].

2) Data integrity: Data integrity is the overall accuracy, completeness, and consistency of data. Data integrity also refers to the safety of data. Message streaming with its originality and without any duplication, reordering, changes, replays are taken care of by a connection-oriented integrity. It also covers the destruction

of data and addresses the denial of service [15]. On the other hand, in a connectionless integrity service, one that handles private messages regardless of any large context, usually protects only from modification of the message. There is a difference in service with and without redemption. If a breach of integrity is encountered, the service may report this breach, and some other mediation like a piece of software or human intervention may require to recover it. In other ways, there are some algorithms to overcome the breach. As for the DSA, if the attacker accessed the data and changed it, the signature verification fails at the receiver node. The output of the verification algorithm will not match the hash of the changed data. Therefore, the receiver can deny the message that the integrity of the message has been violated.

3) Non-repudiation: Non-repudiation prevents either sender or receiver from denying a transmitted message. Hence, the sender and receiver can prove that the message is received and sent by the intended receiver and sender respectively [16]. In this case, the signer only knows the private key and using the private key it creates a unique signature for every given data. Therefore, in the event of any dispute in the future, the recipient can present the data and the digital signature to the third party as proof.

Global Public-Key Components

p prime number

q prime divisor of $p-1$, where $0 < q < p$

$g = h^{(p-1)/q} mod\ p$, where h is any integer with $1 < h < p-1$

such that $h^{(p-1)/q} mod\ p > 1$ (g has order $q\ mod\ p$)

User's Private Key

x is a randomly or pseudo randomly generated integer with $0 < x < q$

User's Public Key

$y = g^x mod\ p$

User's Per- Message Secret Number

k is a randomly or pseudo randomly generated integer with $0 < k < q$

Signing

$r = (g^k mod\ p) mod\ q$

$s = (k^{-1}(SHA-1(M) + xr)) mod\ q$

Verifying

$w = (s')^{-1} mod\ q$

$u1 = ((SHA-1(M'))w) mod\ q$

$u2 = ((r')w) mod\ q$

$v = (((g)^{u1}(y)^{u2}) mod\ p) mod\ q$

TEST: $v = r'$

Fig. 5. The Digital Signature Algorithm (DSA) [12].

C. Hash Algorithms

The hash algorithm converts a data string into a static length numeric string output which is usually smaller than the actual data. Hash algorithms are modeled to be collision-resistant so that different data will have a

low probability of producing the same string. The Message-Digest algorithm 5 (MD5) and the SHA. are the common hash algorithms. To validate the data integrity, the checksums are used in MD5.

Hash algorithms are associated with hash tables to store and recover the data. It converts the key of the data into the hash value and stores in the indexes of the hash table. If an object is inserted in the table, it is added in its empty slot. If it is full, some sort of collision-resistant techniques will be used to discard the new object or replace the old one or store it in a different location using some procedures. To search an object in the hash table, the same procedure is used until it locates the required object. The different versions of SHA algorithms are SHA-1 and SHA-2. Both of the algorithms differ in their construction of the hash table and bit- length of the hash code.

Based on the bit length, the SHA algorithms are classified. The SHA–1 is a 160-bit hash code. The SHA-2 is a group of algorithms that differ in their bit lengths. The SHA-256 and SHA-512 are the most popular in SHA-2. A hash value *h* is generated by a function *H* of the form given by the equation (1) as follows:

$$h = H(M) \tag{1}$$

where *M* is the message and *H(M)* is the fixed-length hash code. The message is appended with the generated hash code and transmitted to the terminal by the host. The terminal recomputes the hash code for authentication. Since the hash function is not confidential, some measures are needed to protect the hash code.

D. SHA-1

SHA-1 is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value. This hash value is known as a message digest. This message digest is usually then rendered as a hexadecimal number which is 40 digits long. For example, the process of file verification explains the SHA-1. In some websites, the files are provided with the SHA-1 hash code on the download page. The user can check the hash code to verify the downloaded file whether it is same as the one intended to download. In this example, consider the same file is downloaded from a different website and it will have a different hash code. Now, the known hash code is compared with the hash code of the file downloaded from a different website. If the two hash codes are different, the file may contain hidden malware, the contents of the file will not be the same, or it may be a corrupted file that causes damage to other files in the disk, etc. It may also mean that one file is older than the other. Thus, a unique hash code will be generated even for a small change.

IV. LIGHTWEIGHT DIGITAL SIGNATURE ALGORITHM

The proposed scheme deals with the generation of signature for a message at the signer and the verification of the signature by the verifier and also gives the details about the link formation. The receiver which verifies the signature can form the link with the sender.

A. Proposed Scheme (Lightweight Digital Signature Algorithm)

Based on the asymmetric cryptographic algorithm, a lightweight digital signature scheme is proposed to secure the information. In this scheme, the signer generates the signature using their private key, appends it to the message's hash code, and sends it to the receiver. The receiver verifies the signature with the signer's public key. Upon successful verification, a link forms between the devices, allowing message transmission. The main steps in this scheme are key and signature generation, signature verification and link formation.

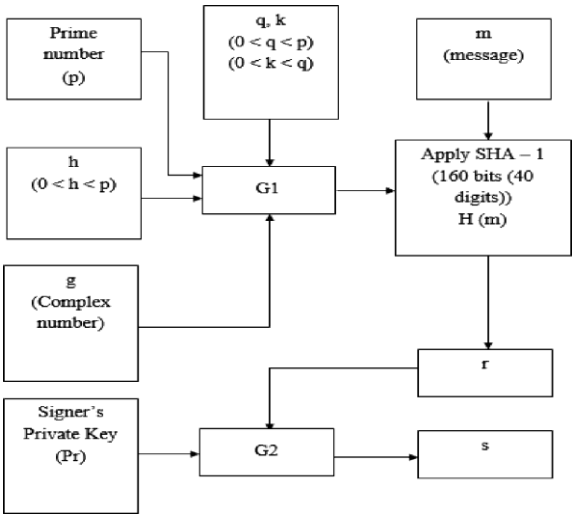


Fig. 6. Signature generation process.

1) Key and Signature Generation: To generate the key pair for every device, the prime number should be chosen. In DSA, the major problem is the usage of large numbers for the signature and verification process. In the proposed scheme, the overhead is reduced by considering a small prime number along with complex numbers concept to introduce secrecy and the Fig.6 shows the signature generation process. The following parameters are considered in the proposed scheme for signature generation and verification. The prime number p is selected from the prime number range. The q is the prime divisor of $p - 1$, where $0 < q < p$ and k is the randomly or pseudo-randomly generated number which is assigned to each message, where $0 < k < q$. The equation $g = h^{(p-1)/q} \, mod \, p$, where h is a real number with $0 < h < p$, gives the complex number that will be attached with the hash code. The Pr is the private key of the signer with $0 < Pr < q$ which is generated randomly.

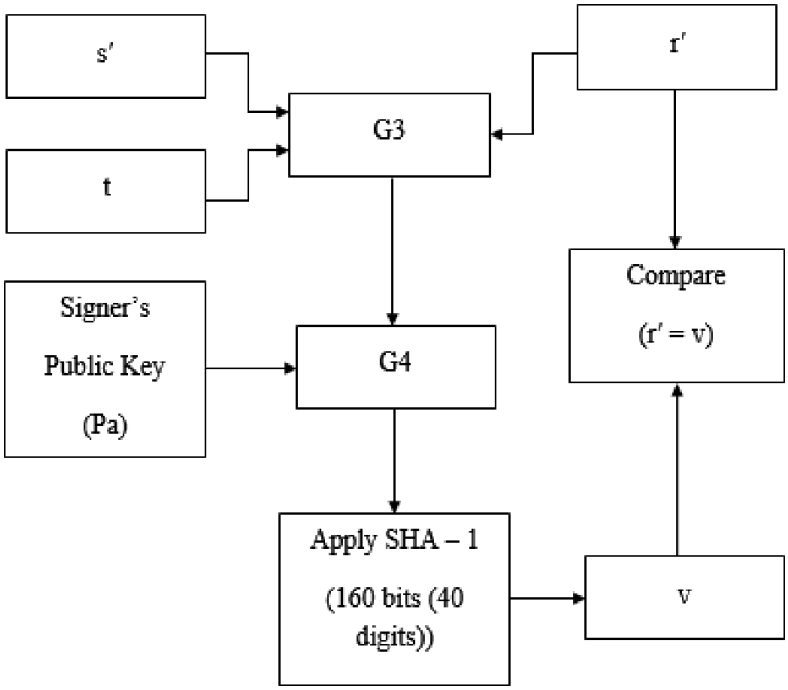


Fig. 7. Signature verification process.

The public key of the signer Pa is given by $Pa = g^k mod \, p$. The integers p, q, and g can be made public which are common to a particular group of nodes. The private and the public keys of the node are usually fixed until the transmission ends. The parameters Pr and k are kept secret and used only for the signature generation. Every time the parameter k should be recomputed for the generation of signature. The secrecy is

introduced by making *g* as a complex number. The pair of numbers *r* and *s* computed using the equations (2) – (4) forms the signature of a message *m*.

$$r1 = \left(g^{k} \bmod p\right) \bmod q \quad (G1) \tag{2}$$

$$r = r1 \parallel H\left(m\right) \tag{3}$$

$$s = k^{-1}\left(H\left(m\right) + \left(Pr * r1\right)\right) \bmod q \quad (G2) \tag{4}$$

In the above equation, k^{-1} is the multiplicative inverse of *k mod q*; that is $(k^{-1}k) mod\ q = 1$ and $0 < k-1 < q$. The output of SHA–1 (*m*) is a 160-bit string output. To use the hash code in computing *s*, the hash string should be converted to an integer. There is a possibility for the *r* and *s* of the signature parameters to be resulting in zero ($r = 0$ or $s = 0$). If any one of the parameters results in zero, the *k* must be re-generated and the signature must be recomputed. Finally, the message is transmitted along with the signature to the recipient or verifier.

2) Signature Verification: The parameters *g*, *p*, and *q* have to be made available at the verifier node along with the public key of the sender node before verifying the signature of the signed message. To find these parameters from the receiving signature, the equations (5) – (7) are used. To carry out the verification, the same algorithm should be made available on both the sender and receiver. The block diagram for the signature verification process is shown in Fig.7. Let the received forms of *r*, and *s* be r', and s' respectively and also consider *Pa* be the signer's public key. Before the signature verification, the verifier has to check for the r' (the first parameter of the signature). To carry out this process, the verifier computes *v* by the equations (8) and (9).

$$t = H\left(m\right)\ s^{-1} \bmod q \tag{5}$$

$$u = r1 * s^{-1} \bmod q \quad (G3) \tag{6}$$

$$Pa = \left(g^{k} \bmod p\right) \tag{7}$$

$$v1 = \left(g^{t} Pa^{k} \bmod p\right) \bmod q \quad (G4) \tag{8}$$

$$v = v1 \parallel H\left(m\right) \tag{9}$$

If the condition $v = r'$ is satisfied, the signature will be verified successfully and the verifier can able to access the message sent by the sender with the private key *Pr*. If the condition is not satisfied, the message may be modified or may be signed incorrectly by the signer. Thus, the message should be considered as an invalid message.

3) Link Formation: The nodes in the network will have their own private and public key and the key allocation to the nodes is carried out by randomizing the prime number every time. The signer node will use its private key and sign the message to produce a signature. The verifier checks the signature against the signer's public key. In the proposed scheme, whenever the receiver verifies the signature, a link will be formed between the sender and the receiver. From the prime number range *p*, every time a prime number *r* per node is assigned.

$$P'' = 1 - P' \tag{10}$$

$$P' = \frac{(p-r) * (p-r-1) * (p-r-2) * \ldots * (p-r-r)}{(p-2-r)! * p^{r}} \tag{11}$$

$$P' = \frac{(p-r)!}{(p-2-r)! * p^{r}} \tag{12}$$

where, P' is the probability of two nodes to form a link by using a prime number r from the prime number range p. From P', the probability to form a link using only one prime number per node from the range is calculated. The P'' represents the probability of two nodes do not form a link.

B. Simulation Results and Discussions

The proposed scheme is analyzed by considering the network with 100 nodes randomly deployed in the area of 50 x 50 (m^2) in the presence of a brute force attack. Nodes are initially deployed at random, followed by the introduction of malicious nodes into the network. Network performance is then evaluated under the impact of these malicious nodes, focusing on connectivity and resilience to node capture [17]. The analysis also examines the network's time complexity. Connectivity assessment reveals the likelihood that two nodes share a common key within the prime number range, shedding light on overall network connectivity and inter-node link establishment. Resilience evaluation quantifies the probability of network compromise as the count of malicious nodes grows. Finally, a comparison is conducted between the proposed scheme and a DSA-based network that excludes complex numbers.

The challenge in the DSA lies in the selection of the prime number to carry out the signature verification process. Thus, in this scheme, the node can be compromised easily by an adversary because it must find only the prime number used to form the link between two nodes. If the prime number gets compromised, an adversary could propagate the attack more rapidly. Yet, the proposed scheme prevents such swift threat expansion, thanks to message signing and verification via complex numbers, combined with randomizing the prime for each link setup. This distinction appears clearly in the comparative resilience analysis graph.

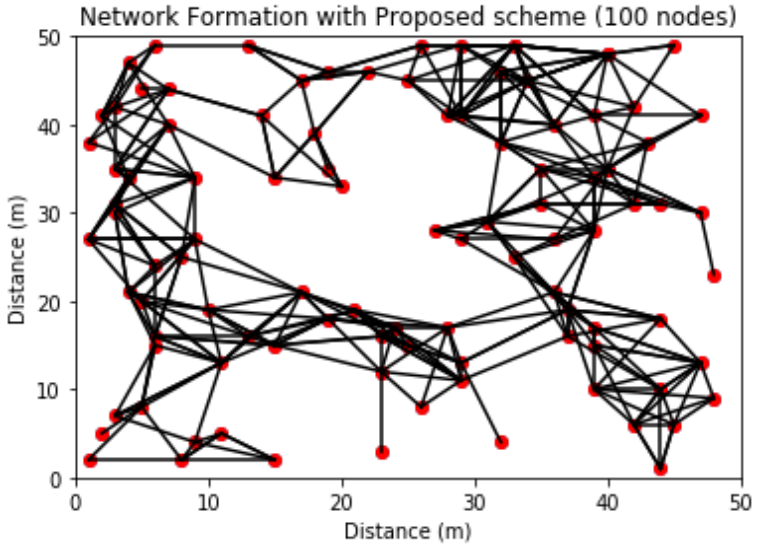


Fig. 8. Network formation with the proposed scheme.

1) Connectivity analysis: Connectivity analysis gives the probability of the two nodes sharing a key in the prime number range. Fig. 8 illustrates the network structure created with the proposed scheme following prime number randomization. From this figure, it's clear that node connectivity remains strong. Randomizing the prime enhances secrecy. Fig. 9 presents the connectivity graph for the proposed scheme, showing that higher prime number ranges reduce link formation probability, though it stays within tolerable limits. As the randomization of prime number selection is done, the lower prime number range shows the highest connectivity. Here, the selection of a prime number from the lower prime number range is easy when compared to the prime number from the highest prime number range. Hence, it is essential to choose the

optimum prime number range. The proposed scheme's key benefit lies in establishing connectivity using lightweight keys, which minimizes memory needs and simplifies the process of choosing the largest prime number.

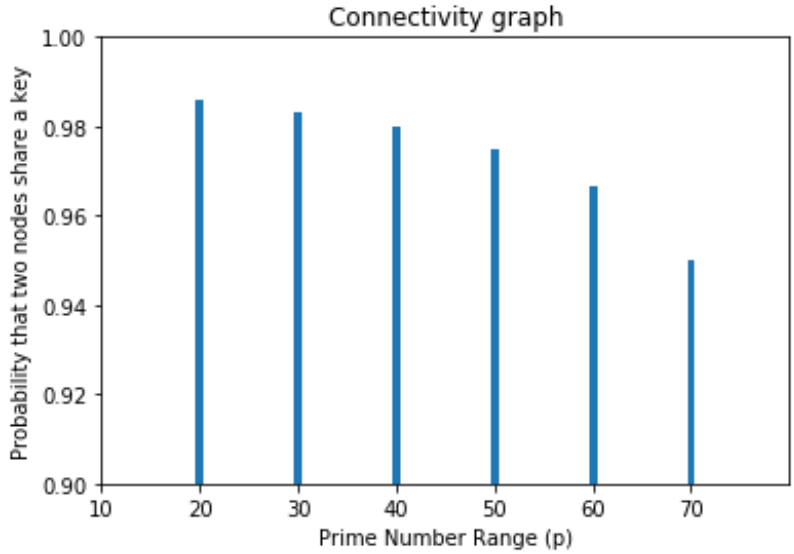


Fig. 9. Connectivity analysis of the proposed scheme

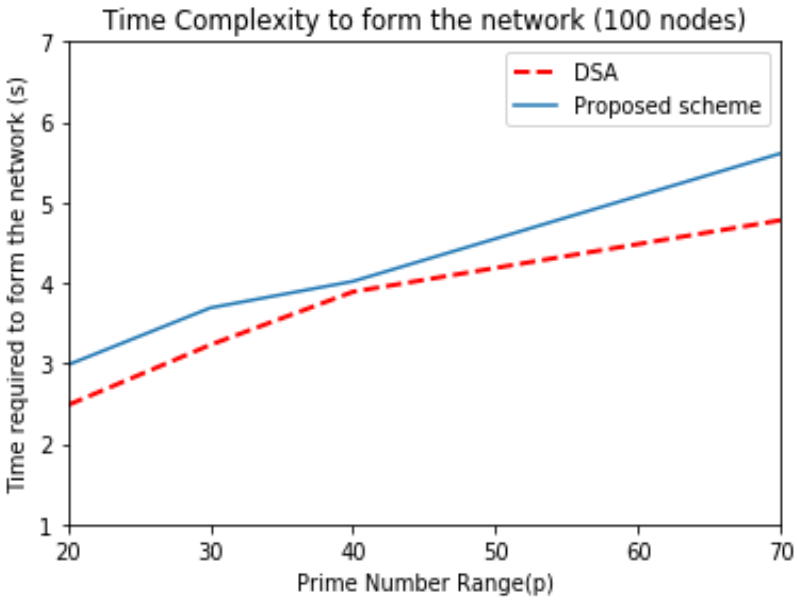


Fig. 10. Time complexity to form the network with different prime range

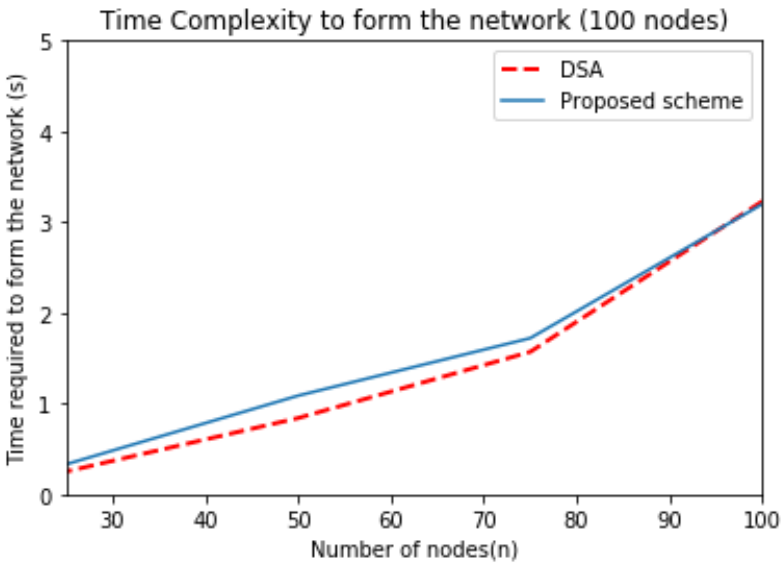


Fig. 11. Time complexity to form the network with different number of nodes

2)Time Complexity: The time consumed during the signature and verification operation of a message for different numbers of values of the prime number range (p) 20, 30, 40, 50, 60 and 70 in case of the proposed method is elucidated by Fig. 10. The time complexity for the different number of nodes and their link formation is also computed and it is shown in Fig. 11. To obtain accurate values, the algorithm was run on an average of 30 times. The proposed scheme takes more time than DSA without complex numbers, yet it delivers superior secrecy. DSA is the process of selecting a prime number and using it in the signature and verification process. Thus, the complexity lies in the selection of prime number. In the proposed method, even with the small prime number, the secrecy can be achieved because of the usage of a complex number in the signature and the verification process. Thus, the proposed method consumes more time when compared to the DSA without complex numbers.

3) Resilience analysis: Resilience analysis measures the likelihood of complete network capture as malicious nodes are added. Malicious nodes are randomly injected into the network across 50 simulation runs to gauge their success in taking over the entire system. Simulations vary the malicious node count, recording capture frequency under brute-force attacks as previously noted. A final comparison contrasts the resilience curves of the proposed scheme against DSA without complex numbers. Fig. 12 depicts the resilient analysis of the proposed scheme for a different prime number range. The Fig. 13- 15 depicts the comparative resilience analysis for the prime number range (p = 20, p = 40, p = 70) between the proposed scheme and the DSA without a complex number concept for the brute force attack with the size of the network being 100. Analysis shows that DSA without complex numbers offers the lowest resilience, getting compromised frequently even with minimal malicious nodes. Fig. 14 reveals that the proposed scheme withstands up to 32 malicious nodes before full compromise, whereas DSA without complex numbers falls completely with just 9 such nodes. Thus, the proposed scheme outperforms the DSA without a complex number and hence it is highly resilient.

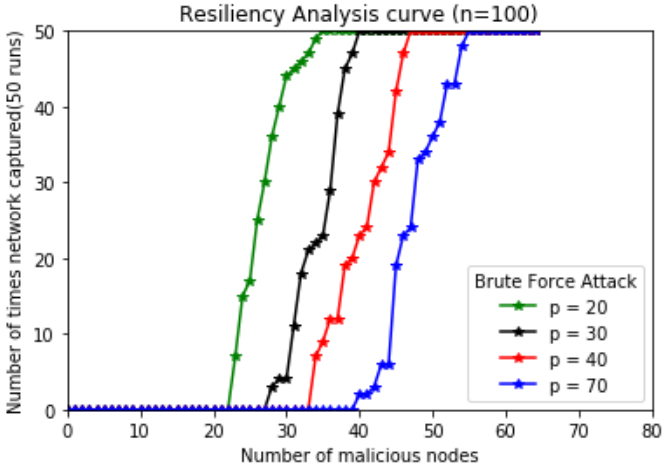


Fig. 12. Resilience analysis of the proposed scheme.

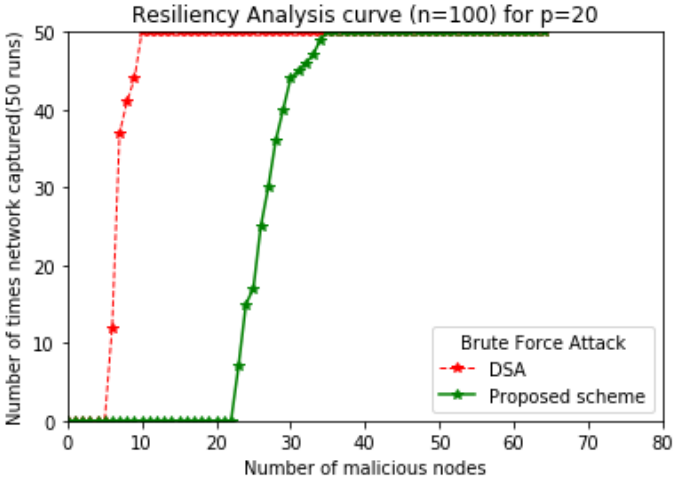


Fig. 13. Comparative resilience analysis for p=20.

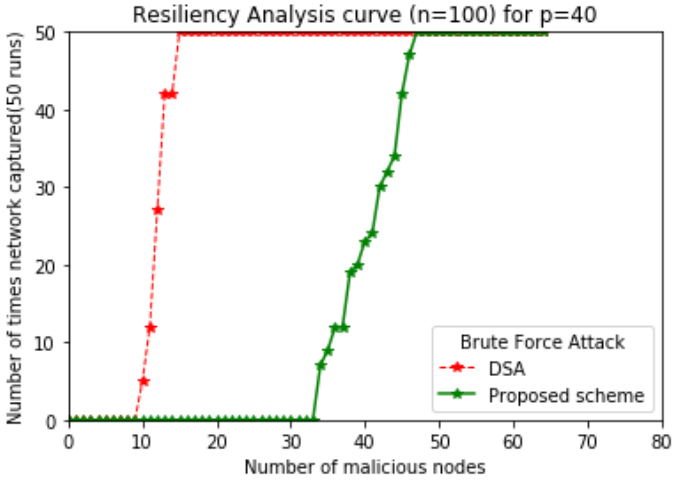


Fig. 14. Comparative resilience analysis for p=40.

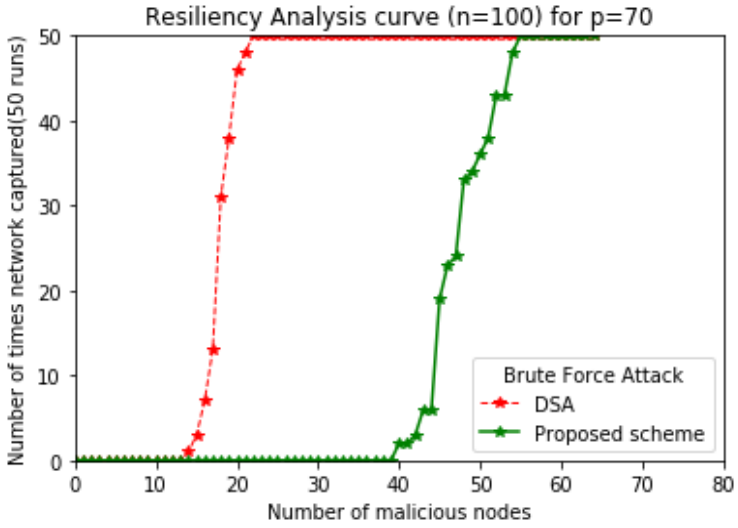


Fig. 15. Comparative resilience analysis for p=70

C. Experimental Results and Discussions

The proposed scheme is experimentally analyzed by considering the network with six raspberry pi nodes in the existence of brute force attack. First, the raspberry pi nodes were installed with the OS and then configured. Once the raspberry pi is configured, the socket is created in every node [18]. Socket formation is a way for two nodes to communicate with each other. A socket (node) listens to a particular port on an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server [19]. Once, the socket is created successfully, the transmitter node can send the required message. Here, according to the proposed method the message is first signed in the transmitter node and then transmitted to the receiver node. The receiver node receives the signature and processes the signature and produces a verification code. If the code is verified with the first parameter of the received signature, the receiver node can able to access the message sent by the transmitter node. Thus, the secure communication is established between two nodes in the network.

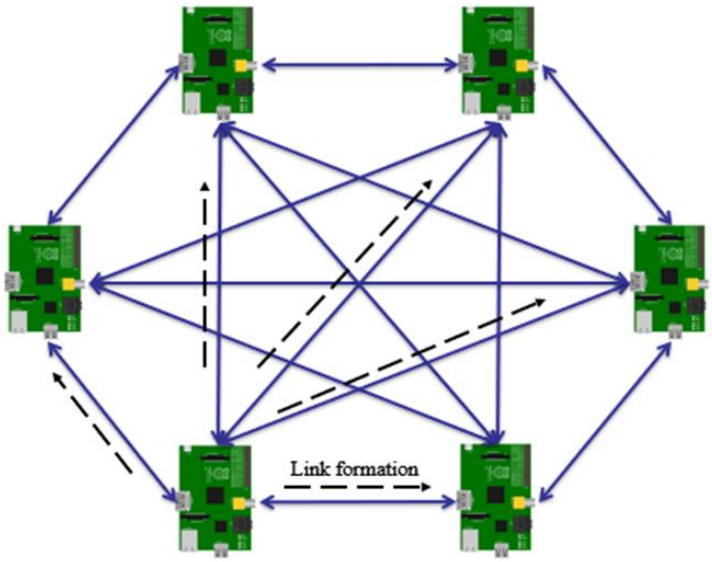


Fig. 16. Mesh network of the proposed scheme

1) Network analysis: The network is modelled using the six raspberry pi nodes under Buster OS platform. With reference to the home automation using IoT, the mesh topology is adopted to the network for experimental evaluation [20]. The Fig. 16 depicts the mesh topology of the network. Here, the six nodes are interconnected with one another using socket programming. In every node, the socket is created using the

same port. Thus, all nodes in the network can reach out one another using the assigned port through the socket. The Fig. 17 depicts the experimental setup of the network. The transmitter node first transmits the signature to the other nodes in the network through the socket. The receiver nodes receive the signature and produces a verification code and verify the code with the signature. If the code got verified, then the node which verified the signature could able to access the message. The Fig. 18 and Fig. 19 depicts the sample output of the proposed scheme's signature and verification process respectively. In the network, the brute force attack is introduced on a node to check out the secrecy property of the algorithm. Considering the hash of the message is known to the attacker, the first four numbers of the signature are taken to carry out the brute force attack. The Fig. 20 depicts the experimental result of the brute force attack.

Fig.17.Experimental setup

.

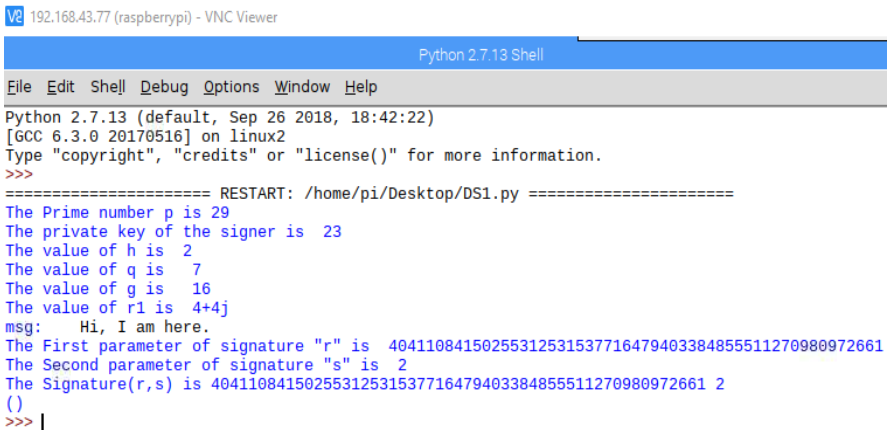


Fig. 18. Sample output of signature process – Experimental result

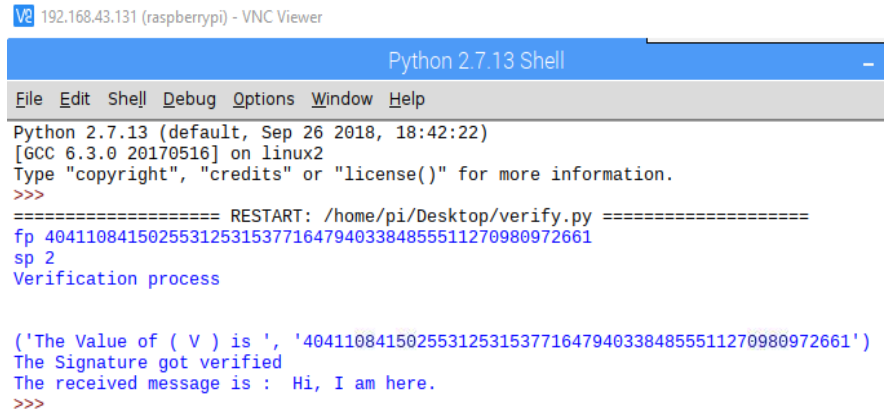


Fig. 19. Sample output of Verification process – Experimental result

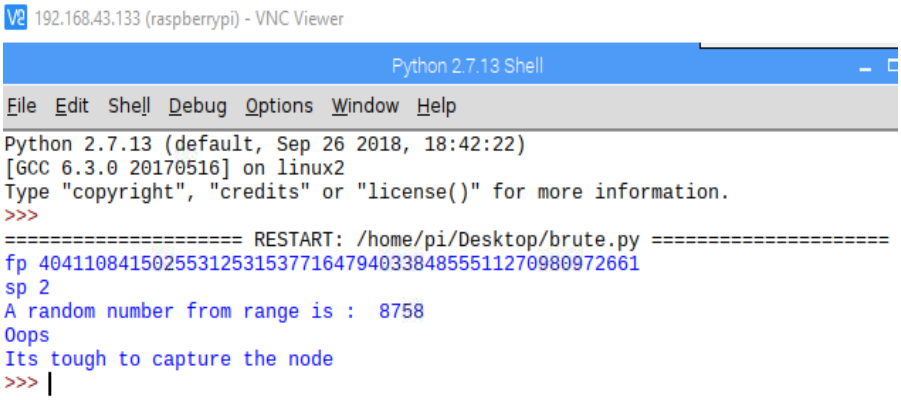


Fig. 20. Sample output of brute force attack – Experimental result

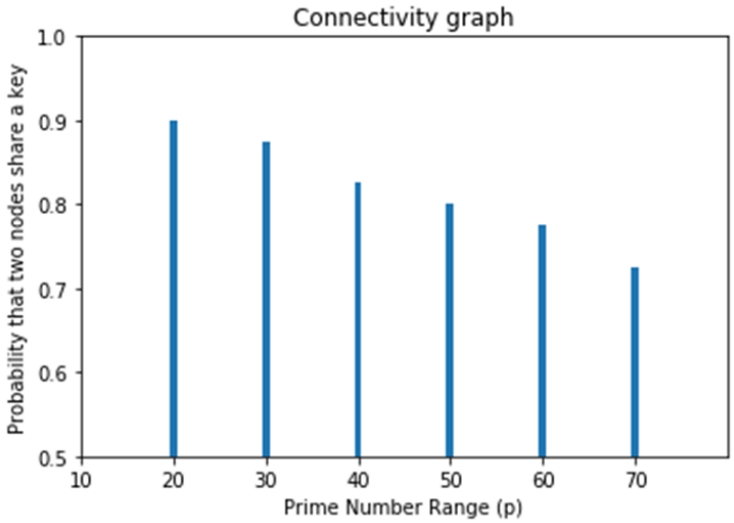


Fig. 21. Connectivity graph – Experimental analysis

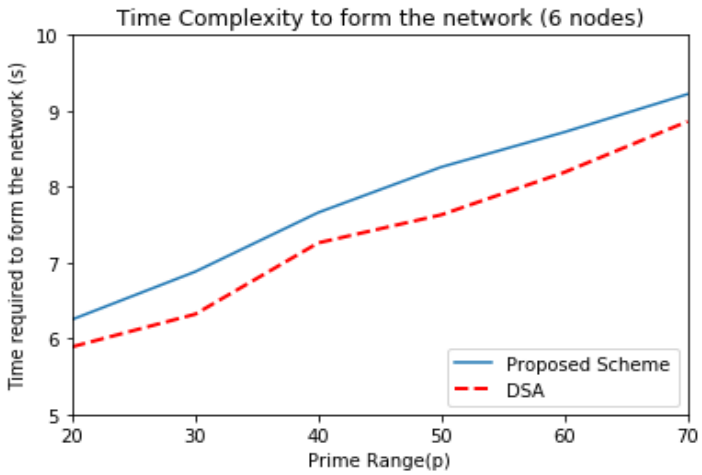


Fig. 22. Time complexity – Experimental analysis

2) Connectivity analysis: Connectivity analysis outlines the odds of two nodes sharing a key within the prime number range. Figure 21 displays the connectivity graph from the proposed scheme's experimental evaluation. Observations from Figure 21 indicate that link formation probability drops as the prime number range expands, yet remains within acceptable bounds. Therefore, it is essential to choose the optimum prime number range. Comparing the Fig. 9 and Fig. 21, it is inferred that the connectivity is less in the experimental analysis. This is because of the real time execution of the proposed algorithm in the network and it depends on the strength of the internet.

3) Time complexity: The Fig. 22 depicts the time consumed during the signature and verification process of a message for different prime number range (p) 20, 30, 40, 50, 60 and 70 in case of experimental analysis of the proposed method. To obtain accurate values, the algorithm was run on an average of 30 times. Fig. 22 indicates that the proposed scheme requires slightly more time than DSA without complex numbers, though it achieves greater secrecy. The usage of complex number in the signature and the verification process consumes more time when compared to the DSA without complex numbers. On comparing the Fig. 10 and Fig. 22, it is inferred that the time required to execute the algorithm is high in the experimental analysis. This is due to the manual execution of the algorithm in every node in the network.

4) Resilience analysis: Resilience analysis quantifies the chances of full network capture as malicious nodes are introduced. One network node is designated malicious to assess its potential to seize the entire system across 50 simulation runs. Fig. 23 illustrates the capture frequency under brute-force attacks. The analysis compares the proposed scheme against DSA without complex numbers, revealing that DSA lacks resilience, suffering frequent compromises unlike the proposed scheme's network.

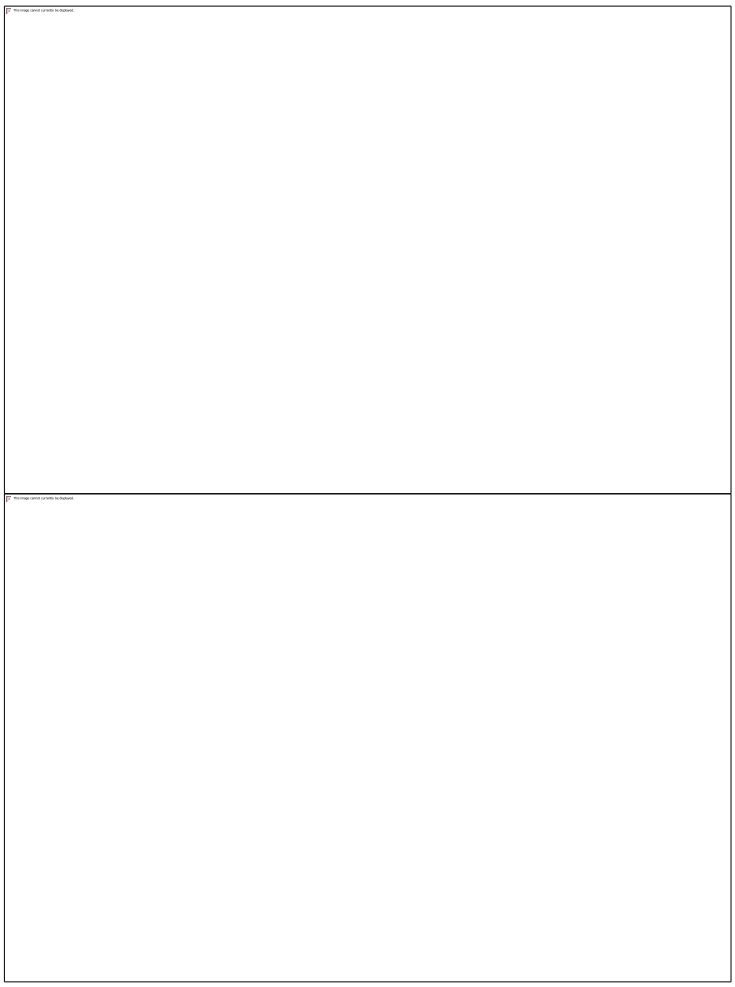


Fig. 23. Comparative resilience analysis for p=20

V. CONCLUSIONS

The IoT devices serve as a key implementation technology for many smart applications. As the Iot devices can be compromised by attackers, the security of the information become a major challenge. Therefore, secrecy for sensitive data and developing solutions against forgery attacks is mandatory. Many cryptographic algorithms were provided in the literature to overcome the attacks and to provide secure communication, but smart devices require lightweight algorithms. Digital signatures are considered a reliable option to ensure the ownership and trustworthiness of devices. Hence, in this work, the lightweight digital signature algorithm was proposed to enhance the resilience against the attacks with lightweight signature when compared to the traditional methods. In signature-based approaches, the major problem lies in the selection of large number for signing and verifying the signature. In the proposed work, the limitation in the traditional approach is overcome by selecting a small prime number and including the complex terms in the computation of signature. In this method, a prime number is randomly chosen from the prime range for each instance. Hence, prime number range should be optimally selected to get better connectivity and resilience. The simulation results of the proposed scheme show that it has better resistance with a lightweight signature against the brute force attack than the traditional method. It is because in the traditional method the resilience increases by selecting a large prime number and the length of the signature also increases, whereas in the proposed method small prime numbers from the prime number range are enough to produce the signature and to achieve resilience against attacks. The results also demonstrate the resilience of the proposed scheme increases with an increase in the prime number range. The experimentation of the proposed scheme using raspberry pi as

nodes also shows that the resilience of the network is better using lightweight signature compared to the traditional methods.

REFERENCES

1. F. Hu. *Security and Privacy in Internet of Things (IoTs): Models, Algorithms, and Implementations*. CRC Press, 2016.
2. T. Zia and A. Zomaya. "Security Issues in Wireless Sensor Networks." In *2006 International Conference on Systems and Networks Communications (ICSNC'06)*, 40–40. IEEE, 2006.
3. M. Kamal and S. Tariq. "Light-Weight Security and Data Provenance for Multi-Hop Internet of Things." *IEEE Access* 6 (2018): 34439–34448.
4. M. Wazid, A. K. Das, V. Odelu, N. Kumar, M. Conti, and M. Jo. "Design of Secure User Authenticated Key Management Protocol for Generic IoT Networks." *IEEE Internet of Things Journal* 5, no. 1 (2017): 269–282.
5. M. A. Mughal, X. Luo, A. Ullah, S. Ullah, and Z. Mahmood. "A Lightweight Digital Signature Based Security Scheme for Human-Centered Internet of Things." *IEEE Access* 6 (2018): 31630–31643.
6. Y. Chen, W. Xu, L. Peng, and H. Zhang. "Light-Weight and Privacy-Preserving Authentication Protocol for Mobile Payments in the Context of IoT." *IEEE Access* 7 (2019): 15210–15221.
7. C. Pu and S. Lim. "A Light-Weight Countermeasure to Forwarding Misbehavior in Wireless Sensor Networks: Design, Analysis, and Evaluation." *IEEE Systems Journal* 12, no. 1 (2016): 834–842.
8. W. Li, H. Song, and F. Zeng. "Policy-Based Secure and Trustworthy Sensing for Internet of Things in Smart Cities." *IEEE Internet of Things Journal* 5, no. 2 (2017): 716–723.
9. R. Giuliano, F. Mazzenga, A. Neri, and A. M. Vegni. "Security Access Protocols in IoT Capillary Networks." *IEEE Internet of Things Journal* 4, no. 3 (2016): 645–657.
10. A. Pinto and R. Costa. "Hash-Chain Based Authentication for IoT Devices and REST Web-Services." In *International Symposium on Ambient Intelligence*, 189–196. Springer, 2016.
11. M. Jahan, M. Rezvani, Q. Zhao, P. S. Roy, K. Sakurai, A. Seneviratne, and S. Jha. "Light Weight Write Mechanism for Cloud Data." *IEEE Transactions on Parallel and Distributed Systems* 29, no. 5 (2017): 1131–1146.
12. W. Stallings. *Cryptography and Network Security*, 4th ed. Pearson Education India, 2006.
13. T. A. Ahanger and A. Aljumah. "Internet of Things: A Comprehensive Study of Security Issues and Defense Mechanisms." *IEEE Access* 7 (2018): 11020–11028.
14. A. Boukerch, L. Xu, and K. El-Khatib. "Trust-Based Security for Wireless Ad Hoc and Sensor Networks." *Computer Communications* 30, no. 11–12 (2007): 2413–2427.
15. M. Frustaci, P. Pace, G. Aloi, and G. Fortino. "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges." *IEEE Internet of Things Journal* 5, no. 4 (2017): 2483–2495.
16. T. A. Ahanger and A. Aljumah. "Internet of Things: A Comprehensive Study of Security Issues and Defense Mechanisms." *IEEE Access* 7 (2018): 11020–11028.
17. K. Rajendiran, R. Sankararajan, and R. Palaniappan. "A Secure Key Predistribution Scheme for WSN Using Elliptic Curve Cryptography." *ETRI Journal* 33, no. 5 (2011): 791–801.
18. D. Ugrenovic and G. Gardasevic. "Performance Analysis of IoT Wireless Sensor Networks for Healthcare Applications." In *The 2nd International Conference on Electrical, Electronic and Computing Engineering (IcE-TRAN)*, 8–11, 2015.
19. V. Kakar. "Secure Communication and Authentication Using Raspberry Pi." In *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 521–525. IEEE, 2016.
20. A. Burg, A. Chattopadhyay, and K.-Y. Lam. "Wireless Communication and Security Issues for Cyber–Physical Systems and the Internet-of-Things." *Proceedings of the IEEE* 106, no. 1 (2017): 38–60.