International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 13 August 2023 **Revised:** 27 September 2023 **Accepted:** 14 October 2023*

An Efficient Secure Group Authenticated Key Agreement Protocol for Wireless Sensor Networks in IoT Environment

Srinivas Kalime*[1], Dr.K.Sagar[2]
[1]Research Scholar,Computer Science and Engineering
University College of Engineering,Osmania University
Hyderabad,Telanagana,India
srinivaskits966@gmail.com
[2]Professor,Computer Science and Engineering
Sreyas Institute of Engineering and Technology
Hyderabad,Telanagana,India
Sagar.k@sreyas.ac.in

Abstract—Internet of Things(IoT) consist of interconnected devices for transmitting and receiving the data over the network. Key management is important for data confidentiality while transmitting in an open network. Even though several key management techniques are feasible to use, still obtaining a key management technique is a challenge with respect to energy and computational cost. The main intention of this work is to discover and overcome the design issues of the existing system and implement a lightweight and secure solution for that issue. The existing system has a fatal security flaw that leads to the unavailability of a complete system which is considered a huge problem in Internet of things. To overcome this issue, an authenticated key management protocol is proposed which deals with the problem of single point of failure and maintains the security properties of the existing system. An authenticated scheme is provided using elliptic curve and hash functions. This scheme also provides client addition, deletion and key freshness. Security analysis and computation complexity has been also discussed. We experimented proposed algorithm and tested with Scyther verification tool. The design overcomes the issues of an existing system by utilizing our scheme in peer to peer network. This network resolves the issue of a single point of failure (SPOF) by distributing the resources and services to the multiple nodes in the network. It will dissolve the problem of SPOF and will increase the reliability and scalability of the IoT system.

Keywords- Authentication, Key agreement, IoT, WSN, Man in the middle attack, Reply attack.

I. INTRODUCTION

Today IoT has expanded upto 12.3 billion connected devices since 2008. It is rapidly growing and now IoT [1] is a part of numerous domains of technology, such as healthcare, smart city, smart energy, home automation etc. IoT is difficultto define as it is very dynamic from its beginning, but can be understood as a network of digital and analog devices provided with unique identifiers (UID) that can exchange data without any human involvement [2]. In many cases, it is seen as an intermediate application, usually a mobile application, which continues to send data to one or more IoT devices [3]. End devices can complete tasks if required and send data back to the application. IoT has expanded the technology to a much more advanced level of integrity, availability, confidentiality,

expansion of device connectivity.

The major concerns of IoT are as follows:

Type of Devices- IoT devices are often classified as

(a). Resourceful devices: These devices are equipped with high processing power and resources.

(b). Constrained devices: Number of smart applications are running on constrained devices; these devices have inadequate resources and limited processing power. They will sense the surroundings, communicate with other devices in the network and act constantly to run the system efficiently.

Security requirement: IoT devices communicate through network by sending and receiving raw and very essential data, so the security of such devices data is very crucial. This networked data is made secure at both the ends by using authentication and this data is encrypted [6]. Authentication, Confidentiality and Integrity are the security requirements included in IoT system. The above security measures can be implemented by using cryptographic algorithms and key management techniques. Cryptographic algorithms are of two types viz., symmetric and asymmetric algorithms; for IoT normally symmetric algorithms are used as they are comparatively light weight to be used for securing data transmission.

- **Key management:** It is an important security characteristic for IoT. Light weight secure key distribution is important for secure and efficient communication. Key distribution mechanisms used in IoT are broadcast, group, shared key distribution as well as node–master architecture. The important factor [1,2] of key management is to reduce the complexity of operations, power consumption of the system and enhance the efficiency and security of the complete system [7,8].

In Dynamic Key Management Scheme, for every session new key is assigned for operation [22]. It does not need revocation or update command to update the key, when the connection between sender and receiver is terminated. In dynamic key management scheme, its observed that there are three ways of generating keys viz., contributory,centralized and distributive.

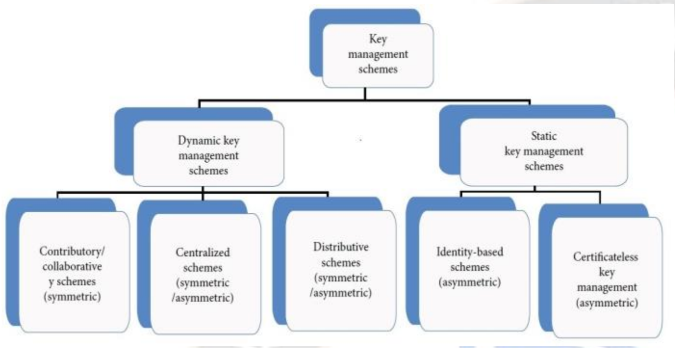

Key management schemes
Dynamic key management schemes
Static key management schemes
Contributory/ collaborative y schemes (symmetric)
Centralized schemes (symmetric /asymmetric)
Distributive schemes (symmetric /asymmetric)
Identity-based schemes (asymmetric)
Certificateless key management (asymmetric)

Figure 1. Classification of KMS algorithm

A. Security Requirements

The Security goals in group communication requires to maintain following security parameters

Confidentiality: Confidentiality in the IoT ensures that any outsider should not have access to the system. It is ensured by using encryption.

Integrity: Integrity in the aspect of group communication refers to the reception of the unaltered message by every group member.

Entity authentication: Entity authentication provides assurance of identity of each member presentin the group.

Non-repudiation: Non-repudiation service provides a guarantee that a member of a group cannot deny sending or receiving a message from the group at a later time.

Availability: Availability assures that the system is available to the group members at all times. It canbe assured by implementing redundant and robust mechanism.

Security and privacy [3] can be achieved by satisfying aforementioned security goals. This can be implemented within a system through cryptographic tools such as encryption, message authentication, digital signature and use of hashing algorithms. Only concern with this tool is that key and keying material should be shared among the parties prior to secure communication.

Group key establishment is an important problem in IoT communication [6], and it is more difficult in IoT environment as it has very limited resources [8]. A naive way to achieve is to rely on cryptographic primitives that allow group entities to establish cryptographic keys within them with a secure and efficient approach [12].

II. AUTHENTICATION AND KEY AGREEMENT SCHEMES

This section contains the brief study of existing authentication schemes for IoT and wireless network environment. In [1], the author presented a systematic literature review to present a comprehensive survey on authentication and key management in IoT. In IoT-based networks, verifying node/user identity requires unique and intrinsic properties satisfied to complete the secure authentication process. Therefore, many protocols are proposed based on various properties such as password-based user authentication [12], Token-based transaction authentication[13],One-time password (OTP) based authentication [14], Location-based authentication [15] and Biometric-based authentication[16] in wireless communication networks such as IoT.

Majid Mumtaz et al. [9] proposed RSA based authentication scheme for a smart IoT environment. In this model, the Public Key Infrastructure (PKI) is used to provide authentication.

Sheetal Karla et.al. [10] proposed ’ECC based mutual authentication system for secure communication between cloud servers’ and embedded devices via HTTP cookies. They claim that the proposed method is resistant to a variety of attacks.It also satisfies all security requirements. But, Saru Kumari et.al. [11] demonstrated how Kalra and Sood’s authentication protocol is sensitive to off-line password guessing attack and insider attack. She also proved that the scheme is vulnerable to many attacks. The gaps identified in the scheme are mutual authentication not being achieved as described in the scheme, the sensor device anonymity not maintained through the scheme, and two parties not agreeing upon the session key computed.

Yang et al. [21] attempted to solve the bottleneck of anonymous credentials and proposed a lightweight authenticated key exchange protocol for IoT applications using a Dynamic Accumulator (DA).

Turkanovic et.al [22] described a scheme for ’WSN based on the IoT notations’ which ensures key agreement and mutual authentication. In this, a lightweight computation based secure key agreement allowed a mobile user with sensor node to negotiate a session key. Although the user never contacts the gateway node, the scheme guarantees the mutual authentication between the sensor node, gateway device and user. Khan et al. [15] proposed a finger print based remote authentication scheme for mobile devices. It has been demonstrated that this scheme is sensitive to desynchronization attacks and user impersonation attack. The user anonymity also not ensured by the scheme.

H.S.Islam et.al. [6] analyzed several problems of Lin’s scheme[10] based on the chaotic maps. Then,he introduced an dynamic identity-based user authentication system for mobile users in 2016. He claimed that it is efficient and robust based on security of extended chaotic maps. Debiao He et.al.[17] introduced a bilinear pairing based anonymous mobile user authentication protocol. The proposed protocol includes a bilinear pairing operation and a hash operation to improve user performance.

Challa et. al.[18] proposed an authentication scheme based on ECC. In this scheme, if both parties mutually authenticated, a valid user may access data collected from a sensing system of IoT application. Jia et.al [23] recently proposed a three-factor key exchange scheme for IoT based on the Challa et. al. (2017)'s [24] authenticated key exchange scheme, which lacks untraceability, anonymity. This scheme is vulnerable to missing smart card attack, impersonation attacks, and password guessing attacks. In the extended security model, the improved three-factor authenticated key exchange scheme secures security. The majority of current authentication [24] and key exchange schemes [25,26] are based on ECC, in which key computation part is scalar-point multiplication. The scalar-point multiplication is significantly more expensive than symmetric encryption/decryption, hash function evaluation, and MAC generation/verification. Furthermore, despite the use of ECC, the majority of schemes fail to achieve user anonymity and mutual authentication[27,28].However, the ECC based encryption is recommended for resource constrained devices like IoT devices to design anonymous and authentication scheme combined with Identity based encryption.

A. Different types of Key management schemes

This section presents study of existing Key management schemes and group Key management schemes for IoT and wireless network environment.

1) Centralized Schemes (Symmetric/Asymmetric): In this scheme, Centralized trusted authority (TA) plays the main role in the process of generating and distributing the unique session key for all the members of a group[4,5]. Due to the dynamic nature and multi-node design of IoT, it is difficult to manage the update operation of the key. The public key of a user is fully or partially certified by the certifying authority(CA). CA is also concerned with the verification of the public key of other users. The key management is unaffordable in the case of IoT devices due to its resource-constrained nodes[6]. Trusted Authority based systems are reliable and efficient than decentralized systems but due to distributed dynamic and resource-constrained nature of IoT,it is unsuitable for implementation[7].

2)Distributive Schemes (Symmetric/Asymmetric): The distributive key management scheme does not use a centralized architecture as it depends on number of entities for the key distribution and management[13]. It is possible to use both variations of cryptographic systems such as symmetric and asymmetric for key generation and management. IoT needs key establishment to be spontaneous during network initialization. In this scheme, trusted authority generates a key for participating node and for its allocation[12].

3) Static Key Management Scheme: In this scheme, the key to be used is generated and used for the overall life cycle of the nodes that are participating in the communication. This scheme uses both symmetric and asymmetric cryptography for mutual agreement and centralized certification authority(CA) services respectively[13].The static key management approach provides a key for the lifetime of a node; whereas in the dynamic approach, a key is assigned for each session. In this scheme, the key is generated once and remains applicable until and unless revoked or updated by certifying authority.

4) Identity-based schemes(Asymmetric): In this scheme, some form of identifier regarding node is used to generate the public key. While generating a private key it depends on TA i.e trusted authority also known as a Private key generator[4]. But there are issues with this scheme such as privacy issues as it uses node focal identities to generate a public key. It always uses public-key cryptography to generate pair of keys. If a PKG is compromised all the messages which are protected using that public-private key are also compromised[7]. This situation makes PKG vulnerable for attacks.

5) Certificate-less key Management(Asymmetric): In this scheme Certificate-less public key cryptography is used to guarantee the authenticity of public keys[5]. This scheme depends on a trusted third party(TTP) who possesses the master key. In this regard, Certificate-less public key cryptography is the same as an identity-based scheme[30], but it does not suffer from the problem of key escrow property(also known as fair cryptosystem). Therefore it can be seen as an intermediate cryptographic scheme between regular Public Key Cryptography (PKC) and identity-based PKC[29]. The threshold cryptography schemes are used to overcome the limitations and constraints on the key generation.

III. BASIC CONCEPTS

A. Elliptic curve cryptography (ECC)

The Elliptic curve cryptography (ECC) is coined by Neil Koblitz and Victor Miller[5,6].For easy calculation, the arithmetic operations are performed in the finite fields F. Software and hardware applications may employ different finite fields. Software applications use prime fields(F_p),while hardware applications use binary fields(F_{2^m}).

The Weierstrass equation,

$$v^2=u^3+au+b(mod p) \qquad (1)$$

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 13 August 2023 **Revised:** 27 September 2023 **Accepted:** 14 October 2023*

where constant integers a and b are less than a prime number *p* that hold following equation,

$$4a^3 + 27b^2 \neq 0(mod p) \quad (2)$$

An elliptic curve *E(a, b)* defined over the prime field *(Fp)* is defined by selecting −1 and 0 for values of *a* and *b* in equation 1 respectively. Then, the elliptic curve represents the equation $v^2 = u^3 - u$ shown in Fig.2

B. *Elliptic Curve Discrete Logarithm Problem*

The ECDLP is proved computational hard problem on elliptic curves, therefore, the security of the cryptography protocols designed using ECC are secure based on hardness of the ECDLP. It is to find *k* for given values of *P* and *Q* in the equation *Q = kP*. Finding the value of *k* is computationally hard problem.

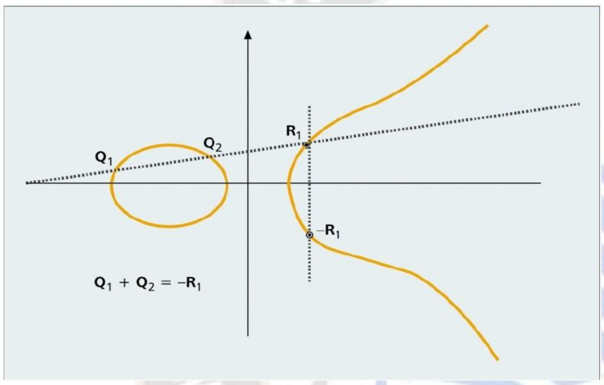


Figure 2. Elliptic Curve: v2 = u3 − u

C. *Hash Function*

A hash function *H(M)* produces a fixed length hash h from a variable length message. The length of the hash *h* fixed regardless of the length of the *M*, even if it is a terabyte. The hash value *h*, marked as one-way property, cannot be used to find the message *M*. Reverse engineering is impossible in this case. It is defined as *h=H(M).* For every different *M,* it outputs the different hash h. It helps to identify any changes to the message M in the transmission through the network. If any content of the message *M* is changed, then hash *h = H(M)* of the message also changes. The adversary can't compute the M message from the message hash h. The forward process is easy, but reverse is hard, this is the reason to call it as"one-way function". Therefore, one-way functions are used for information security in various domains.

The hash function should satisfy one-way function (OWF)property and property of collision resistance. The collision property is, for two different messages *M1, M2,* the same output hash output h should be created. The collision property states that *M1 ≠ M2; H(M1) = H(M2)*

SHA(Secure Hash Algorithm),MD (Message Digests), RIPEMD (RIPE Message Digest), and Whirlpool are some examples of hash functions[?]. The SHA algorithms are most preferred hash functions most extensively used nowadays. Table I shows the properties of the various SHA algorithms.

TABLE I. COMPARISON OF SHA ALGORITHMS (IN BITS)

Algorithm	**Msg.Length**	**Block size**	**Word size**	**Hash**
SHA-1	2^{64-1}	512	32	160
SHA-256	2^{64-1}	512	32	256
SHA-384	2^{128-1}	1024	64	384
SHA-512	2^{128-1}	1024	64	512

IV. PROPOSED AUTHENTICATED GROUP KEY AGREEMENT FOR IOT ENVIRONMENT

In this section, the proposed system is discussed with concerning its design parameters, algorithm, implementation, and performance. The existing systems suffers from the problem of availability as the gateway is a Single Point of Failure (SPOF)[31]. A SPOF is a fundamental problem in the design, functionality, or development of systems, or characteristic that poses a threat because it could result in a scenario in which a single malfunction or fault causes the entire system to crash. Depending on the interdependencies involved in the failure and the location of the failure[31]. To overcome the issue of SPOF we designed the proposed system using peer to peer approach. Before presenting key agreement between node peer and server peer, we present the authentication algorithm.

A. *Proposed Method for Authentication*

Proposed solution for Authentication ECDSA with SHA−512 using NIST-192-P. The ECDSA algorithm with SHA-512 and NIST-192-P elliptic curve can be summarized in the following steps:

1) *Key Generation:*

∗ Generate a random private key, *sk* $\in [1, n-1]$, n is the order of the EC.

∗ Compute the corresponding public key *pk*, as the scalar multiplication of the EC base point *G* with generated private key*(sk)*.
$pk = sk * G$

2) *Signature Generation:*

∗ Hash the message 'msg' using the SHA-512 algorithm to obtain a hash value 'hash'.

∗ Choose a random integer *k* $\in [1, n-1]$, n is the order of the EC.

∗ Compute the EC point *P* as the scalar multiplication of the base point G with *k*:

$P = k * G.$

- $*$ Calculate the value r as the x-axis coordinate of the point P modulo n:
  $r = (P.x \bmod n).$
- $*$ Calculate the value s:
  $s = k^{-1} * (hash + r * sk) \bmod n$
  where k^{-1} is the mod inverse of $k \bmod n$.
- $*$ Signature pair: (r, s).

3) *Signature Verification:*

- $*$ Verify: $r, s \in [1, n - 1].$
- $*$ Hash the original message' msg' using SHA-512 to obtain the hash value 'hash'.
- $*$ Compute w as the mod inverse of s mod n.
- $*$ Calculate: $u1 = (hash * w) \bmod n.$
- $*$ Calculate: $u2 = (r * w) \bmod n.$
- $*$ Compute the EC point *'X'* as the point addition of $(u_1 * G)$ and $(u_2 * pk)$: $X = u_1 * G + u_2 * pk.$
- $*$ Compute x-axis coordinate: $r' = X.x \bmod n.$
- $*$ Calculate the signature validation result by comparing Whether $r' == r$ otherwise, it is invalid.

4) *Signature verification proof:*

The equation behind the recovering of the point X, calculated during the signature verification, can be transformed by replacing the pk with $sk * G$ as follows:

$* X = (hash * w) * G + (r * w) * pk$

$* X = (hash * w) * G + (r * w) * sk * G$

$* X = (hash + r * sk) * w * G$

$*$ If $s = [(k^{-1} * (hash + r * sk)) \bmod n]$

$*$ Here $w = s^{-1} \bmod n$

$*$ *i.e* $w = ([(k^{-1} * (hash + r * sk)) \bmod n])^{-1}$

$* w = (k * [(hash + r * sk) \bmod n])^{-1}$

$*$ Replace w *in* X then $X = (hash + r * sk) * k * [(hash + r * sk) \bmod n])^{-1} * G$ *that implies* $X = k * G = P.$

$*$ Compute *'X'* coordinate *i.e* $r' = X.x \bmod n = P.x \bmod n = r$

$*$ *Hence* $r' = r.$

B. *Proposed Group key aggreement using peer to peer approach*

A peer-to-peer system [32]. is a network architecture that allows peers to share system resources, processing capacity, and data storage without the need for a centralized server. Participants in a peer-to-peer(P2P) network can serve as both client and server simultaneously. They can be directly accessed by other nodes without using intermediary entities [31]. In the proposed system, peers are composed of two roles viz., Sponsor peer and Node peer. Sponsor peer will work as a serving authority to the Node peers.

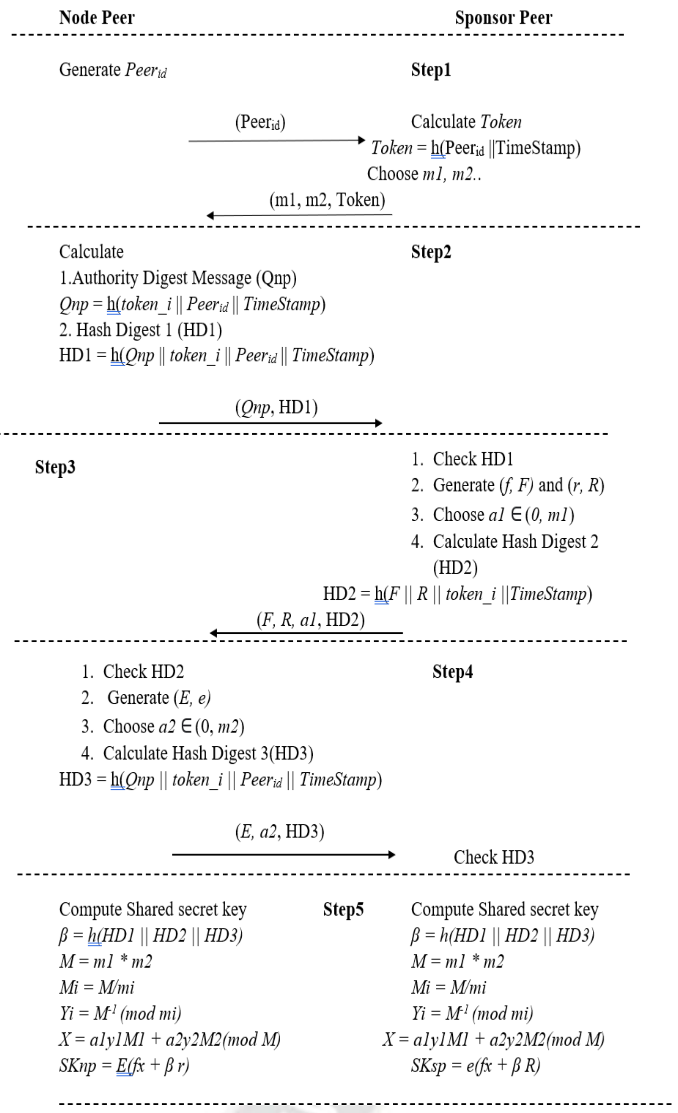

Node Peer
Sponsor Peer
Generate $Peer_{id}$
Step1
$(Peer_{id})$
Calculate *Token*
$Token = h(Peer_{id} \| TimeStamp)$
Choose *m1, m2..*
(m1, m2, Token)
Calculate
1.Authority Digest Message (Qnp)
$Qnp = h(token\_i \| Peer_{id} \| TimeStamp)$
2. Hash Digest 1 (HD1)
$HD1 = h(Qnp \| token\_i \| Peer_{id} \| TimeStamp)$
Step2
$(Qnp, HD1)$
Step3
1. Check HD1
2. Generate (f, F) and (r, R)
3. Choose $a1 \in (0, m1)$
4. Calculate Hash Digest 2 (HD2)
$HD2 = h(F \| R \| token\_i \| TimeStamp)$
$(F, R, a1, HD2)$
1. Check HD2
2. Generate (E, e)
3. Choose $a2 \in (0, m2)$
4. Calculate Hash Digest 3(HD3)
$HD3 = h(Qnp \| token\_i \| Peer_{id} \| TimeStamp)$
Step4
$(E, a2, HD3)$
Check HD3
Compute Shared secret key
$\beta = h(HD1 \| HD2 \| HD3)$
$M = m1 * m2$
$Mi = M/mi$
$Yi = M^{-1} (mod\ mi)$
$X = a1y1M1 + a2y2M2(mod\ M)$
$SKnp = E(fx + \beta r)$
Step5
Compute Shared secret key
$\beta = h(HD1 \| HD2 \| HD3)$
$M = m1 * m2$
$Mi = M/mi$
$Yi = M^{-1} (mod\ mi)$
$X = a1y1M1 + a2y2M2(mod\ M)$
$SKsp = e(fx + \beta R)$

Figure 3. Algorithm: Peer to Peer key establishment

–Step 1:

1. Node Peer will generate a unique identification number $Peer_id$ required for registration. This $Peer_id$ is sent to the Sponsor Peer.
2. Sponsor Peer will calculate a unique token (code$Token_i$) = *hash* ($Peer_id$ ||*Timestamp*) for node peer.
3. Sponsor peer will choose two prime number viz., *m1* and *m2*
4. At the end of this stage it will send $Token_i$, *m1* and *m2* to node peer.

–Step 2:

1. Node Peer calculates two parameters viz., Authority Digest Message and first message digest as follows
   a. Authority Digest Message Q_np = *hash* ($Token_i$|| $Peer_id$ || *TimeStamp*)

b. $HashDigestHD1=hash(Q_np \| Token_i \| Peer_id \| TimeStamp)$
2. Node Peer sends above parameters to Sponsor Peer.

–Step 3:

1. Checks the hash digest received from node peer by comparing with the calculated hash.
2. Generates two pairs of public and private keys for further operations.
3. Chooses a random number from the range of *m1*.
4. Calculates hash digest message $HD2 = hash(F \| R \| Token_i \| TimeStamp)$
5. Sends the *F, R, a1, HD2* to the node peer.

–Step4:

1. Checks the hash digest received from node peer by comparing with the calculated hash.
2. Generates a public and private key pair for further operations.
3. Selects a random number from the range of *m2*.
4. Calculates hash digest message $HD3 = hash(Q_np \| Token_i \| Peer_id \| TimeStamp)$
5. Sends the *E, a2, HD3* to the Sponsor peer.

–Step5: Both Node peer and Sponsor peer will calculate parameters required to generate a common shared key.

1. Calculates $\beta = hash(HD1 \| HD2 \| HD3)$
2. $M = m1 * m2$
3. $M_i = M / m_i$
4. $Y_i = M^{-1} \pmod{m_i}$
5. $X = a_1y_1M_1 + a_2y_2M_2 \pmod{M}$
6. $SK_np = E(fx + \beta * r)$
7. At sponsor peer side shared key will be
   $SK_sp = e(fx + \beta . R)$

V. SECURITY ANALYSIS

Security analysis of protocols entails a detailed review of the design, implementation, and deployment of a protocol to detect possible vulnerabilities and assure its resilience against assaults. The study tries to find problems that might affect the security objectives of the protocol, such as confidentiality, integrity, authenticity, and availability.

A. Security Analysis Tool

Security analysis tools are software programs or frameworks that aid in the examination and assessment of the security of computer systems, networks, applications, or protocols. These programs automate numerous security analysis activities, offering useful insights into possible vulnerabilities, flaws, or misconfigurations. The following are types of security analysis tools available for formal analysis and verification of the protocol:

--*Automated Validation of Internet Security Protocols and Applications:* AVISPA is a complete toolkit for automating the examination and testing of security procedures. It is suited for a broad variety of protocol analysis jobs since it supports a large srange of security features and attack models. AVISPA is made up of various components that work together to produce a robust protocol analysis environment:

-*SPAN:* Stochastic Protocol Analyzer is a model checking tool that employs formal approaches to validate security protocols. It may identify weaknesses, vulnerabilities, and logical problems in protocol designs.

– *OFMC:* The Optimized Finite Model Checker is a tool for analyzing security protocols that use symbolic model checking. It can handle complicated protocols with many sessions and tests for different security features.

–*TA4SP:* The Timed Automata Analyzer for Security Protocols analyses timed security protocols by taking time-related features and delays into account. It aids in the evaluation of procedures with time limits.

–*SATMC:* The Satisfiability Model Checker is a model checking tool that validates security characteristics. Inconsistencies, inaccessible states, and possible attacks in protocols may be detected.

– *Scyther:* Scyther is a sophisticated tool for analyzing and verifying security mechanisms. It utilizes symbolic modeling approaches to concentrate on the formal verification of security features. Scyther has an easy-to-use command-line interface, making it accessible to security researchers and practitioners.

∗ *Protocol Modeling:* Scyther's straightforward language enables for the development and modeling of security protocols. Users may specify the protocol's participants, responsibilities, messages exchanged and security features.

∗ *Automatic Verification:* Scyther automates the validation of security procedures against predefined security attributes. It examines a variety of security features, such as secrecy, authentication, freshness, and others. Scyther uses formal methods and symbolic analysis tools to investigate all conceivable protocol executions to identify potential vulnerabilities or breaches of security characteristics.

∗ *Counter example Generation:* Scyther may produce counter-examples that demonstrate an attack or violation when a security property is breached. These counterexamples aid in understanding the security vulnerability and help protocol debugging and development.

∗ *Visualization:* Scyther includes visualization tools for displaying protocol models, hypothetical attacks, and counterexamples. This graphical depiction assists in comprehending protocol behavior, finding vulnerabilities, and presenting analysis results. Scyther accepts input in a variety of forms, including Scyther Language, High-Level Protocol Specification Language (HLPSL), and Protocol Composition Logic (PCL). It is extremely configurable, enabling customers to define desired security attributes and alter the analytic settings to meet their needs

B. *Authentication and Key Exchange Scyther verification*

Scyther is a formal verification tool used for analyzing security protocols. It allows for the specification and verification of cryptographic protocols, including authentication protocols.

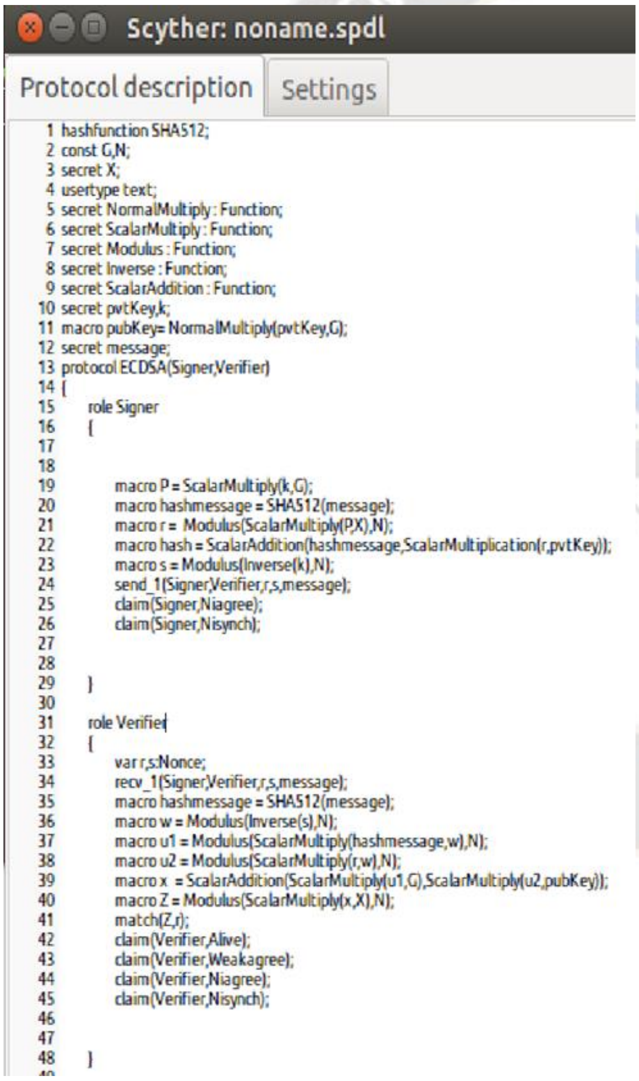

```
hashfunction SHA512;
const G,N;
secret X;
usertype text;
secret NormalMultiply : Function;
secret ScalarMultiply : Function;
secret Modulus : Function;
secret Inverse : Function;
secret ScalarAddition : Function;
secret pvtKey,k;
macro pubKey= NormalMultiply(pvtKey,G);
secret message;
protocol ECDSA(Signer,Verifier)
{
    role Signer
    {


        macro P = ScalarMultiply(k,G);
        macro hashmessage = SHA512(message);
        macro r =  Modulus(ScalarMultiply(P,X),N);
        macro hash = ScalarAddition(hashmessage,ScalarMultiplication(r,pvtKey));
        macro s = Modulus(Inverse(k),N);
        send_1(Signer,Verifier,r,s,message);
        claim(Signer,Niagree);
        claim(Signer,Nisynch);


    }

    role Verifier
    {
        var r,s:Nonce;
        recv_1(Signer,Verifier,r,s,message);
        macro hashmessage = SHA512(message);
        macro w = Modulus(Inverse(s),N);
        macro u1 = Modulus(ScalarMultiply(hashmessage,w),N);
        macro u2 = Modulus(ScalarMultiply(r,w),N);
        macro x  = ScalarAddition(ScalarMultiply(u1,G),ScalarMultiply(u2,pubKey));
        macro Z = Modulus(ScalarMultiply(x,X),N);
        match(Z,r);
        claim(Verifier,Alive);
        claim(Verifier,Weakagree);
        claim(Verifier,Niagree);
        claim(Verifier,Nisynch);


    }
```

Figure 4. Scyther Code for ECDSA

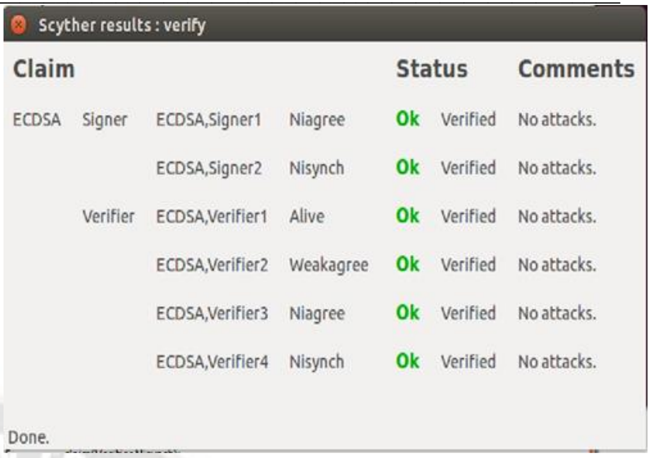

```
Claim                                              Status        Comments

ECDSA  Signer    ECDSA,Signer1    Niagree          Ok  Verified  No attacks.
                 ECDSA,Signer2    Nisynch          Ok  Verified  No attacks.
       Verifier  ECDSA,Verifier1  Alive            Ok  Verified  No attacks.
                 ECDSA,Verifier2  Weakagree        Ok  Verified  No attacks.
                 ECDSA,Verifier3  Niagree          Ok  Verified  No attacks.
                 ECDSA,Verifier4  Nisynch          Ok  Verified  No attacks.

Done.
```

Figure 5. Scyther Verification of ECDSA

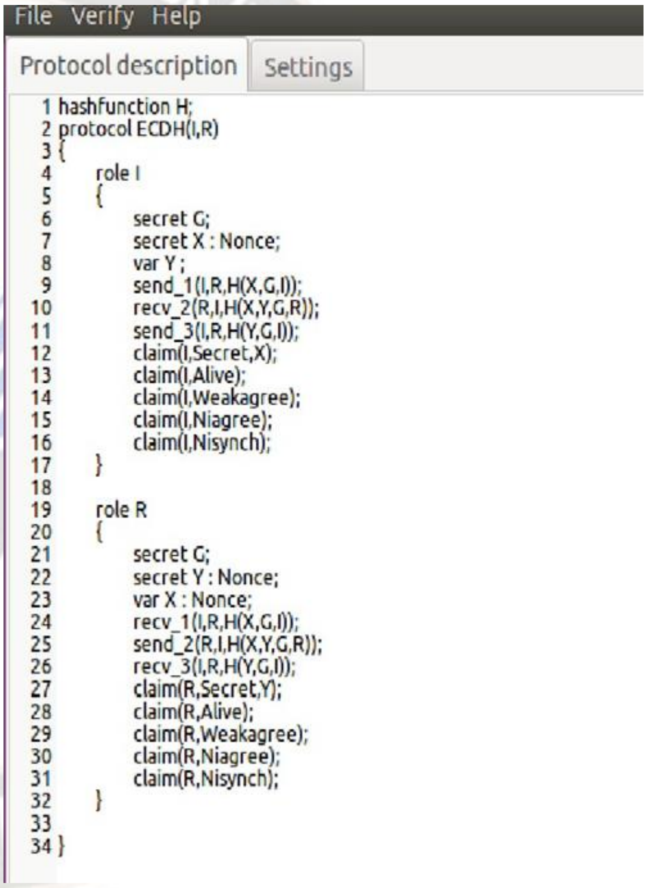
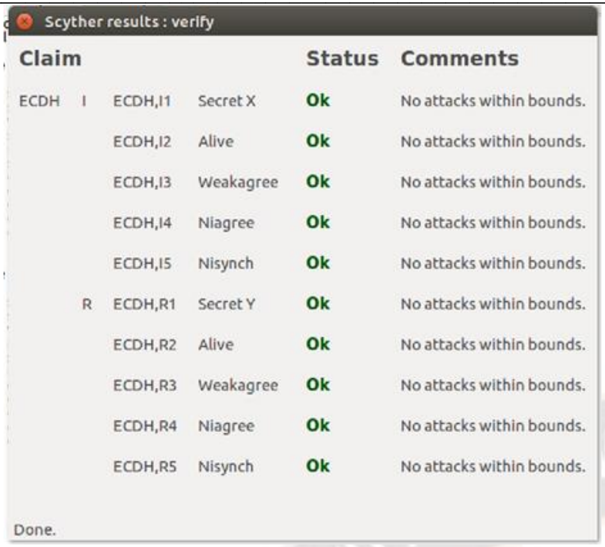

```
hashfunction H;
protocol ECDH(I,R)
{
    role I
    {
        secret G;
        secret X : Nonce;
        var Y ;
        send_1(I,R,H(X,G,I));
        recv_2(R,I,H(X,Y,G,R));
        send_3(I,R,H(Y,G,I));
        claim(I,Secret,X);
        claim(I,Alive);
        claim(I,Weakagree);
        claim(I,Niagree);
        claim(I,Nisynch);
    }

    role R
    {
        secret G;
        secret Y : Nonce;
        var X : Nonce;
        recv_1(I,R,H(X,G,I));
        send_2(R,I,H(X,Y,G,R));
        recv_3(I,R,H(Y,G,I));
        claim(R,Secret,Y);
        claim(R,Alive);
        claim(R,Weakagree);
        claim(R,Niagree);
        claim(R,Nisynch);
    }

}
```

Figure 6: Scyther Code for ECDH



Claim				Status	Comments
ECDH	I	ECDH,I1	Secret X	Ok	No attacks within bounds.
		ECDH,I2	Alive	Ok	No attacks within bounds.
		ECDH,I3	Weakagree	Ok	No attacks within bounds.
		ECDH,I4	Niagree	Ok	No attacks within bounds.
		ECDH,I5	Nisynch	Ok	No attacks within bounds.
	R	ECDH,R1	Secret Y	Ok	No attacks within bounds.
		ECDH,R2	Alive	Ok	No attacks within bounds.
		ECDH,R3	Weakagree	Ok	No attacks within bounds.
		ECDH,R4	Niagree	Ok	No attacks within bounds.
		ECDH,R5	Nisynch	Ok	No attacks within bounds.

Done.

Figure7: Scyther Verification for ECDH

The above figures describe the protocol using a formal modeling language, define the roles of the participants, specify the messages exchanged, and define the security properties to be verified. The tool then performs an automated analysis to check the protocol for potential security vulnerabilities, such as authentication failures, replay attacks, or key disclosure.

The analysis proves our proposed authentication and group key agreement protocol is resistant to standard attacks.

TABLE II: OUR METHOD *VS* OTHER EXISTING METHODS

Method	**Techniques used**	**P$_1$**	**P$_2$**	**P$_3$**	**P$_4$**	**P$_5$**
[2]	ECC	X	✓	X	X	X
[4]	ECC+Hash	X	X	X	X	X
[7]	ECC+Hash	✓	✓	X	✓	X
[32]	ECC+Hash	X	✓	X	✓	X
[25]	ECC+Hash	X	✓	✓	✓	X
[23]	ECC+Hash	✓	X	X	X	X
Proposed method	ECC+Hash	✓	✓	✓	✓	✓

In this table, P1-MIM attack, P2-Replay attack,
P3- Impersonation attack, P4-Message Integrity attack,
P5-Traceability attack. ✓- Resistant, X - Non-resistant.

VI. CONCLUSION

The primary goal of this work is to discover and overcome the design issues of the existing system and implement a lightweight and secure solution for that issue. The existing system has a fatal security flaw that leads to the unavailability of a complete system which is considered a huge problem in information security. To overcome this issue, a authenticated key management protocol is proposed which deals with the problem of Single Point of Failure (SPOF) and maintains the security properties of the existing system. a key management protocol deals with three interrelated problems such as device authentication and session key management. The challenge in developing such a protocol to resolve these concerns is making sure that the protocol is lightweight. It means that the protocol is made up of cryptographic primitives that are computationally lightweight for devices with limited resources. To address the above issues, a lightweight key management protocol using a symmetric key approach is proposed in this paper. Our method been tested with security analyser tools such as Scyther and it is resistant to existing attacks.

REFERENCES

[1] Patruni Muralidhara Rao, B.D. Deebak, A comprehensive survey on authentication and secure key management in internet of things: Challenges, countermeasures, and future directions, Ad Hoc Networks 146 (2023) 103159.
[2] Bashar A Alohali, Vassilios G Vassilakis, Ioannis D Moscholios, and Michael D Logothetis. A secure scheme for group communication of wireless iot devices. In 2018 11th International Symposium on Communication Systems, Networks Digital Signal Processing (CSNDSP), pages 1–6. IEEE, 2018.
[3] Anshul Anand, Mauro Conti, Pallavi Kaliyar, and Chhagan Lal. Tare: Topology adaptive re-keying scheme for secure group communication in iot networks. Wireless Networks, 26(4):2449–2463, 2020.
[4] Ali Azougaghe, Zaid Kartit, Mustapha Hedabou, Mostafa Belkasmi, and Mohamed El Marraki. An efficient algorithm for data security in cloud storage. In 2015 15th International Conference on Intelligent Systems Design and Applications (ISDA), pages 421–427, 2015.
[5] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard pkcs 1. In Annual International Cryptology Conference, pages 1–12. Springer, 1998.
[6] Bob Briscoe. Marks: Zero side effect multicast key management using arbitrarily revealed key sequences. In International Workshop on Networked Group Communication, pages 301–320. Springer, 1999.
[7] Joseph Bugeja, Bahtijar Vogel, Andreas Jacobsson, and Rimpu Varshney. Iotsm: an end-to-end security model for iot ecosystems. In 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pages 267–272. IEEE, 2019.
[8] Asit Chakraborti, Syed Obaid Amin, Aytac Azgin, Satyajayant Misra, and Ravishankar Ravindran. Using icn slicing framework to build an iot edge network. In Proceedings of the 5th ACM Conference on Information-Centric Networking, pages 214–215, 2018.
[9] Shehzad Ashraf Chaudhry, Khalid Yahya, Fadi Al Turjman, and Ming-Hour Yang. A secure and reliable device access control

scheme for iot based sensor cloud systems. IEEE Access, 8:139244–139254, 2020.

[10] Cas JF Cremers. Unbounded verification, falsification, and characterization of security protocols by pattern refinement. In Proceedings of the 15th ACM conference on Computer and communications security, pages 119–128, 2008.

[11] A. N. Tentu, A review on evolution of symmetric key block ciphers and their applications, IETE Journal of Education 61(1) (2020) 34–46.

[12] A. Singh, A. N.Tentu, V. C. Venkaiah, "A dynamic key management paradigm for secure wireless ad hoc network communications," International Journal of Information and Computer Security, vol. 14, no. 3/4, pp. 380–402, 2021.

[13] Tentu, A. N., Venkaiah, V. C., Prasad, V. K. (2018). CRT based multi-secret sharing schemes: revisited. International Journal of Security and Networks, 13(1), 1-9.

[14] Tentu, A. N., Raju, K., Venkaiah, V. (2019). Cryptanalysis of a group key transfer protocol: Generalization and countermeasures. Journal of Combinatorics System Sciences, 44, 269-283.

[15] Teklay Gebremichael, Ulf Jennehag, and Mikael Gidlund. Lightweight IoT group key establishment scheme using one-way accumulator. In 2018 International Symposium on Networks, Computers and Communications (ISNCC), pages 1–7. IEEE, 2018.

[16] R. Khan, P. Kumar, D. N. K. Jayakody, and M. Liyanage, "A survey on security and privacy of 5G technologies: Potential solutions, recent advancements, and future directions," IEEE Commun. Surveys Tuts., vol. 22, no. 1, pp. 196–248, 1st Quart., 2020, doi:10.1109/COMST.2019.2933899.

[17] A. Ahad, M. Tahir, and K.-L.-A. Yau, "5G-based smart healthcare network: Architecture, taxonomy, challenges and future research directions," IEEE Access, vol. 7, pp. 100747–100762, 2019.

[18] A. Braeken, M. Liyanage, P. Kumar, and J. Murphy, "Novel 5G authentication protocol to improve the resistance against active attacks and malicious serving networks," IEEE Access, vol. 7, pp. 64040–64052, 2019.

[19] I. Ahmad, T. Kumar, M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov, "Overview of 5G security challenges and solutions," IEEE Commun. Standards Mag., vol. 2, no. 1, pp. 36–43, Mar. 2018.

[20] Mag., vol. 2, no. 1, pp. 36–43, Mar. 2018. 20 P. Gandotra and R. K. Jha, "A survey on green communication and security challenges in 5G wireless communication networks," J. Netw.Comput. Appl., vol. 96, pp. 39–61, Oct. 2017.

[21] Z. Tian, Y. Sun, S. Su, M. Li, X. Du, and M. Guizani, "Automated attack and defense framework for 5G security on physical and logical layers," 2019, arXiv:1902.04009. [Online]. Available:

[22] S. Challa, M. Wazid, A. K. Das, N. Kumar, A. G. Reddy, E.-J. Yoon, and K.-Y. Yoo, "Secure signature based authenticated key establishment scheme for future IoT applications," IEEE Access, vol. 5, pp. 3028–3043, 2017.

[23] S. Challa, M. Wazid, A. K. Das, and M. K. Khan, "Authentication protocols for implantable medical devices: Taxonomy, analysis and future directions," IEEE Consum. Electron. Mag., vol. 7, no. 1, pp. 57–65, Jan. 2018.

[24] Lu K, Qian Y, Guizani M, Chen HH (2008) A framework for a distributed key management scheme in heterogeneous wireless sensor networks. IEEE Trans Wirel Commun 7(2):639–647 34.

[25] X. Fan and G. Gong (2015) LPKM: A lightweight polynomial-based key management protocol for distributed wireless sensor networks," In proceeding of International Conference on Ad Hoc Networks, pp. 180-195.

[26] Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. (2013a). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645-1660.

[27] Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. (2013b). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645-1660.

[28] Ko, I. Y., Ko, H. G., Molina, A. J., Kwon, J. H. (2016). SoIoT: Toward a user-centric IoT- based service framework. ACM Transactions on Internet Technology (TOIT), 16(2), 1-21.

[29] Mohammed Nafi, Mohamed-Lamine Messai, Samia Bouzefrane, Mawloud Omar, IFKMS: Inverse Function-based Key Management Scheme for IoT networks, Journal of Information Security and Applications Volume 71, December 2022, 103370.

[30] Jha, S.; Jha, N.; Prashar, D.; Ahmad, S.; Alouffi, B.; Alharbi, A. Integrated IoT-Based Secure and Efficient Key Management Framework Using Hashgraphs for Autonomous Vehicles to Ensure Road Safety. Sensors 2022, 22, 2529.

[31] Mirvaziri, Hamid and Hosseini, Rahim, A novel method for key establishment based on symmetric cryptography in hierarchical wireless sensor networks, Wireless Personal Communications, vol.112, pp.2373– 2391, 2020.

[32] Jiang, Liaoliang and Li, Tong and Li, Xuan and Atiquzzaman, Mohammed and Ahmad, Haseeb and Wang, Xianmin, Anonymous communication via anonymous identity-based encryption and its application in IoT, Wireless Communications and Mobile Computing, vol-18, 2018