Computers, Materials & Continua
DOI:10.32604/cmc.2021.014233
Article

Tech Science Press

IPv6 Cryptographically Generated Address: Analysis, Optimization and Protection

Amjed Sid Ahmed[1,*], Rosilah Hassan[2], Faizan Qamar[3] and Mazhar Malik[1]

[1]Department of Computing and Information Technology, Global College of Engineering and Technology, Ruwi, 112, Sultanate of Oman
[2]Faculty of Information Science and Technology, Center for Cyber Security, Universiti Kebangsaan Malaysia, Bangi, 43600, Malaysia
[3]Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600, Bangi, Malaysia
[*]Corresponding Author: Amjed Sid Ahmed. Email: amjed@gcet.edu.om
Received: 08 September 2020; Accepted: 24 November 2020

Abstract: In networking, one major difficulty that nodes suffer from is the need for their addresses to be generated and verified without relying on a third party or public authorized servers. To resolve this issue, the use of self-certifying addresses have become a highly popular and standardized method, of which Cryptographically Generated Addresses (CGA) is a prime example. CGA was primarily designed to deter the theft of IPv6 addresses by binding the generated address to a public key to prove address ownership. Even though the CGA technique is highly effective, this method is still subject to several vulnerabilities with respect to security, in addition to certain limitations in its performance. In this study, the authors present an intensive systematic review of the literature to explore the technical specifications of CGA, its challenges, and existing proposals to enhance the protocol. Given that CGA generation is a time-consuming process, this limitation has hampered the application of CGA in mobile environments where nodes have limited energy and storage. Fulfilling Hash2 conditions in CGA is the heaviest and most time-consuming part of SEND. To improve the performance of CGA, we replaced the Secure Hash Algorithm (SHA1) with the Message Digest (MD5) hash function. Furthermore, this study also analyzes the possible methods through which a CGA could be attacked. In conducting this analysis, Denial-of-Service (DoS) attacks were identified as the main method of attack toward the CGA verification process, which compromise and threaten the privacy of CGA. Therefore, we propose some modifications to the CGA standard verification algorithm to mitigate DoS attacks and to make CGA more security conscious.

Keywords: IPv6; GCA; SEND; DoS attacks; RSA; SHA-1

1 Introduction

In the present era, a huge increase in the number of hosts on the Internet is observed; thus, communication and networking experts expect Internet protocol version 4 (IPv4) to be

replaced by IPv6 [1–3]. The IPv6 suite primary protocol is a neighbor discovery protocol (NDP), which is considered as a replacement for the address resolution protocol (ARP) function in IPv4 [4]. The NDP protocol performs an array of functions (definite router functions) that are related to nonrouter or host-specified functions. Moreover, the NDP protocol performs multiple tasks, including node examination, duplicate address detection, stateless address auto configuration (SLACC) [5,6], and address solving and redirection to appropriate routers [7].

IPv6 has existed for 16 years, and most organizations have begun transitioning from IPv4 to IPv6. However, though some organizations have moved to IPv6, many have yet to start their transition [8]. This delay is due to several reasons, one of which is related to IPv6 security [9–11]. NDP internal working mechanisms were developed with the expectation that nodes would be linked with other trusted nodes. However, these assumptions were not observed in real-time implementation [12]. Airports, coffee shops, and other places with a wireless environment are not secure, and anyone can join and threaten the link easily. Owing to this lack of security, the NDP is vulnerable to denial of service (DoS) attacks, which may cause devices to crash [13–16]. The NDP can experience various attacks, such as neighbor advertisements, neighbor solicitations, and numerous frequently occurring threats owing to the trusted assumption of NDP internal working mechanisms. For instance, the duplicate address detection (DAD) attack is a common NDP problem [17,18] owing to the mandatory usage of an address duplication procedure while the NDP configures itself using a SLACC process [19,20]. Such attacks are a threat to many sectors, such as banking and e-commerce, where recovery costs for devices and network operations are considerable. To deal with NDP security issues, the Internet Engineering Task Force (IETF) proposed an updated version of the NDP called the secure neighbor discovery (SEND) protocol [21,22]. The role of the SEND protocol is to ensure address ownership validity while excluding malicious attacks and router authorization procedures [23]. As an updated version of the NDP, the SEND protocol contains four new features, that is, a cryptographically generated address (CGA), an RSA, two ICMPv6 messages, and a timestamp (TS) [24,25]. The CGA is the core of the SEND protocol, which is designed to deter the stealing of IPv6 addresses by binding the generated address to a public key to verify address ownership. Although the CGA technique is highly effective, it remains subject to several security vulnerabilities in addition to certain performance limitations. In this study, the authors modify a standard CGA by proposing a new method for improving CGA performance and security. The need for additional IP addresses with the exponential growth of the number of devices connected to the Internet is evident. Thus, transition to IPv6 is progressing, which will soon completely replace IPv4. The main objective of this research is to highlight and address IPv6 vulnerabilities that threaten devices.

1.1 Contribution

The CGA technology employs an addressing technique of fixed-size addresses with the help of a cryptographic hash function for an address owner's public key. This technique enables owners to confirm ownership of addresses by mapping between addresses and owners using a public/private key pair [26,27]. The contribution of this study is threefold. (i) This study explores CGA technical specifications, deficiencies, and security vulnerabilities. (ii) This study conducts a survey regarding the researchers' proposal to improve CGA performance. (iii) This study introduces two models to improve CGA performance and security, namely, the CGA-Lighter and Locked-CGA models, respectively.

1.2 Paper Organization

The rest of this paper is organized as follows. Section 2 explains the CGA specifications, and Section 3 describes the CGA deficiencies along with various types of CPA attacks and solutions. Section 4 elucidates the methodology and results, and Section 5 discusses the conclusion. Finally, the limitations and future research directions are presented in Section 6.

2 CGA Specifications

In networking, the use of IPv6 improves self-certified addresses with the SEND protocol. The CGA generates a standard 64-bit address with notable symbols, as shown by [26]. Moreover, it can undergo self-verification without the need for a public-key infrastructure to create an IPv6 address. CGA addressing was initially proposed by [28] as a childproofing certification for mobile IPv6 and further improved by [29]. In addition, [30] recommended an alternative process using the "SUVC" concept. Subsequently, [31] demonstrated the process in the real world.

In the IPv6 address format, special semantics and several parameters are used in the interface identifier. The first parameter set is comprised of "u" and "g" bits, which are located in the seventh and eighth bits in the interface identifier. The combination of u = g = 1 is kept unused for other purposes, which can be used in the CGA, as suggested in [31]. The other value in the interface identifier is the security parameter "sec," which is a 3-bit user-defined parameter. In a CGA, this parameter is used in relation to a hash extension, which is an essential notation used in CGA generation, as presented in Tab. 1.

Table 1: Notations CGA generation

Data	Notation	Length
Subnet prefix	SP	64-bit
Interface identifier	Interface ID	64-bit
Public-key	K_pub	Variable-length
Private-key	K_priv	Variable-length
Digital signature	Sign	Variable-length
Modifier	m	128-bit
Collision count	CC	8-bit
Security parameter	sec	3-bit
u,g flags	u, g	1-bit each
Subnet prefix	SP	64-bit
Interface identifier	Interface ID	64-bit
Public-key	K_pub	Variable-length
Private-key	K_priv	Variable-length
Digital signature	Sign	Variable-length
Modifier	M	128-bit

The CGA uses a hash extension method, which is empowered by the sec security parameter. This method includes a linear hash extension by enforcing several 16-sec bits to zero, which is referred to as Hash2. Its main purpose is to improve CGA security. To generate addresses,

a computer must fulfill a number of parameters, such as the hash extension, which could slow the process of address creation if used with a big sec value.

2.1 Creation of Addresses

The following are the steps to produce an IPv6 address that uses CGA [32]:

a. To adjust the modifier randomly to a 128-bit value, select the security value and initialize the collision counter.
b. Add the modifier with 64 + 8 zero bits and the present public-key. Then, run the hash algorithm on these joined character strings. The outcome is Hash2 with 112 bits on the far-left.
c. The 16_sec bits have to be matched with the bits on the far-left with zero. If all are zero, proceed to Step D. Otherwise, increment the modifier and return to Step B.
d. Add the modifier with subnet notation, present public-key, and count of collisions. Then, run this modifier with the hash algorithm on these joined character strings. The outcome is Hash1 with 64 bits on the far-left side.
e. The two special bits, *i.e.,* u and g, are preset to obtain an identifier for an interface on Hash1 to 1 and 3 bits on the far-left to sec-6. A 128-bit IPv6 address is obtained by adding the subnet notation and interface identifier.
f. In a scenario where the IP address has conflict with machines in the same subnetwork, increment the count of collisions and return to Step E. This process avoids three initial conflicts. If it still exists, it halts the process and forwards the error.

2.2 Authentication of Addresses

a. Check and confirm whether the conflict counter is 0. If it is 1 or 2 and the subnet notation is also the same, then CGA authentication automatically becomes unsuccessful.
b. Add the modifier with the subnet notation, the conflict count, and public-key. Then, run the H algorithm on the combined character strings. The outcome is Hash1 with 64 bits on the far-left.
c. The Hash1 to the address interface identifier is to be equated.
d. Alter the two special bits, u and g. The initial 3 bits on the left side are overlooked. In this scenario, the 64-bit values contrast, and the CGA authentication automatically becomes unsuccessful.
e. Add the modifier with 64 + 8 zero bits and preset public-key. Then, run the H algorithm on these combined character strings. The outcome is Hash2 with 112 bits on the left side.
f. The security feature is obtained by the sec 3-bit on the leftmost side of the address interface identifier. Then, equate 0 with 16_sec bits on the left side of Hash2. If the bits are not zero, the CGA authentication will be unsuccessful. Otherwise, the verification will be successful if sec = 0 verification never fails at this step. Fig. 1 illustrates the CGA creation and authentication process [33].

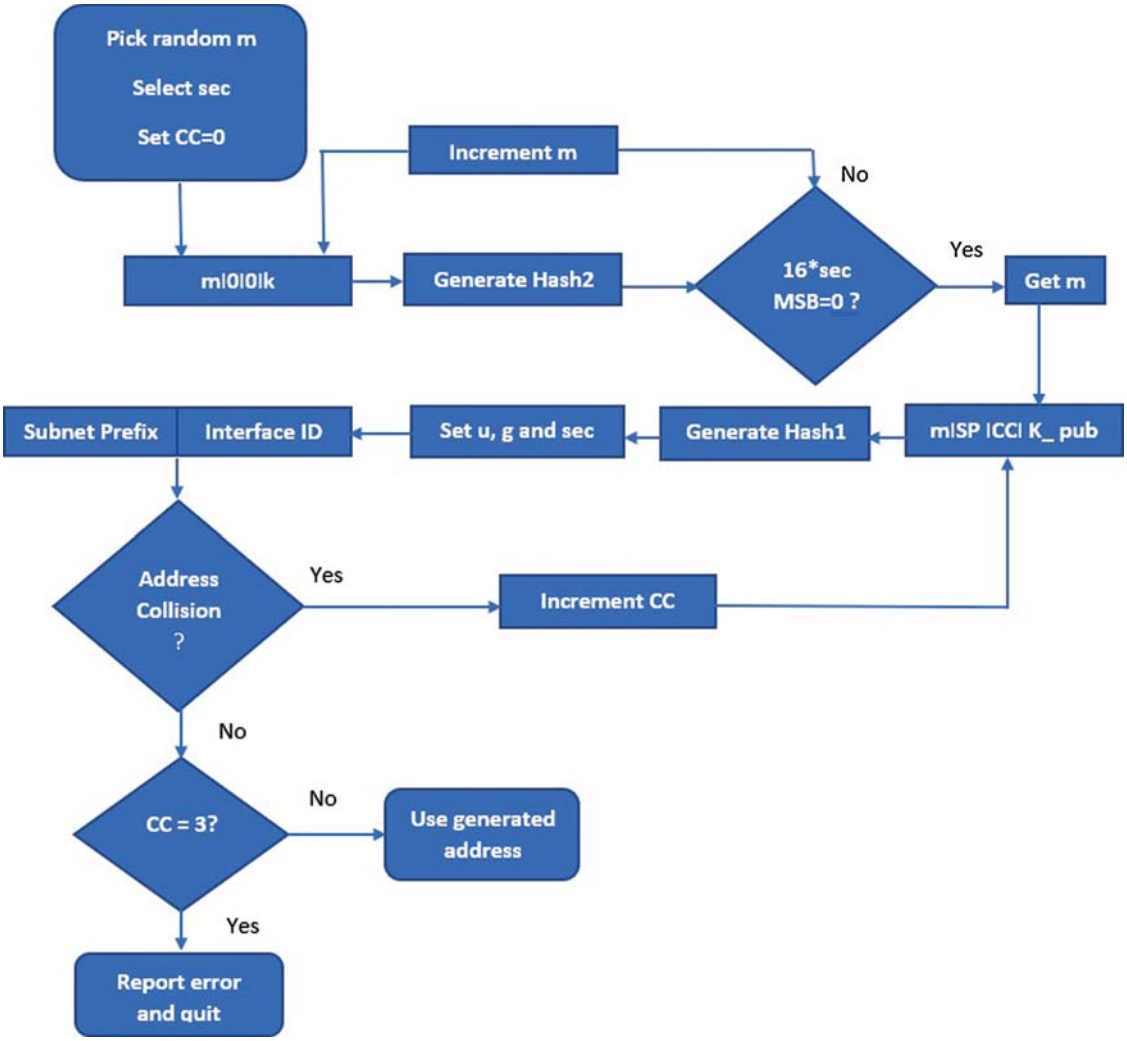

Figure 1: CGA creation and authentication process

3 CGA Deficiencies

The CGA security technique is quite promising in the IPv6 protocol. Nevertheless, it still has some limitations and disadvantages. One of the limitations is that CGA generates addresses with relatively higher computational time [26,34]. Moreover, although CGA provides security to IPv6, it is still vulnerable to threats, and it is not a complete solution for NDP security [27]. For instance, CGA does not assure that the address provided is for a node. Attackers use this drawback by compromising a new and valid CGA address, which is created with its own public key. This new address also captures the messages of Neighbor Discovery (ND) by which attackers can alter the CGA parameters of the sender. This limitation is causing failure in the CGA verification process at the receiver's end. Thus, the communication between the sender and the receiver becomes insecure. A DAD DoS Attack can be conducted by an attacker in various ways, For instance, it disallows a new node from joining the link. The attackers can also use the techniques wherein the CGA parameters and signatures are copied and issued with a Neighbor Advertisement (NA) message, which contains similar security parameters. Through this process, the attackers prevent

the CGA address for all nodes from being configured and from being attached to a local link. In another example, attackers continue the verification process. This action ensures that the node is always busy verifying valid and invalid messages of CGA.

The time required to accomplish the Hash2 demand is stated in Tab. 2 [35] and the outcome performance is matched [36]. The table shows that effective addressing is impossible with an increase in safety parameter sec.

Table 2: CGA generation time for different sec values

Specification of setup	sec = 0	sec = 1	sec = 2	sec = 3
Pentium 4.3 GHz, memory 1 GB Linux (Kernel 2.4)	15.57 μs	just over 0.1 s	100 s	more than 200 h
Machine with moderate processing power	n/a	1 min	16 days)	n/a
A modern PC (AMD64)	n/a	0.2 s	3.2 h	24 years

3.1 CGA Attacks

In this section, we discuss various attacks, such as (1) discovery of an alternative key pair hashing of a victim's node address, (2) detection of a victim node's private key, (3) a global time/memory tradeoff (TMTO) attack, (4) DoS attacks against the CGA verification process, and (5) CGA privacy implication. Attackers use the aforementioned methods to compromise the security of the CGA node [37].

3.1.1 Discovery of an Alternative Key Pair Hashing of a Victim's Node Address (Second Preimage Attacks)

In such attacks, an attacker discovers a victim's alternative key pair hashing address, and the success of the attack may depend on the hash function security properties. That is, an attacker will attempt to break the preimage hash function resistance. According to the RFC 3972 CGA standards, this is vulnerable to collision attacks when used in SHA-1 [38]. RFC 4982 is used to analyze the implications of attacks for the hash function and to propose implementation supporting multiple hash algorithms.

3.1.2 Detection of a Victim Node's Private Key

In this case, an attacker uses a private key in lieu of a public key, copies the CGA, and forges its signature.

3.1.3 Global TMTO Attacks

A CGA is extremely vulnerable to global TMTO attacks, as shown in [27]. In such attacks, an attacker performs a search for hash collisions or a match to numerous addresses using the interface IDs of its public key(s) created from an extensive precomputed database.

3.1.4 DoS Attacks Against the CGA Verification Process

In the CGA verification process, DoS attacks can be executed in specific steps against the DAD and verification of CGA parameters [14].

a) DoS Attacks Against DAD-CGA

According to RFC 3756, IPv6 is highly susceptible to DoS attacks in the DAD algorithm. When DAD is applied to a tentative address by a victim's node, in a reply, an attacker will respond as if the address is in use. As a result, the victim will not be able to configure itself and join a network.

b) DoS Attacks by Replaying the Sender CGA Parameters

A CGA is susceptible to repeated attacks when an attacker tries to steal and store a victim's node messages. Moreover, an attacker can repeat attacks by using a sender's CGA parameters. Generally, this process calculates the Hash1 required by a CGA-enabled receiver, which involves the verification of a sender's interface ID. This verification is required to send the CGA parameters from a sender to a receiver. If a modification is executed for the CGA parameters by an attacker, then Hash1 will fail. If failure exceeds more than two, then the verification process will fail. Thus, it will interrupt the attacker communication between a CGA-enabled sender and receiver.

c) DoS Attacks to Kill a CGA Node

An attacker can keep a node busy with the verification process and send high-frequency signed valid or invalid CGA messages across a network. This technique is a type of DoS attack applied to the request–response protocol and not specific to CGAs.

3.1.5 CGA Privacy Implication

CGA generation requires high computational complexity. If an acceptable CGA is generated once, then it will be continuously used in the subnet, thereby becoming highly susceptible to privacy-related attacks. Utilization of the same address repeatedly over a long period of time increases threats, and an attacker can track and violate a user's online privacy in devices such as cellphones, laptops, and so on.

3.2 Solutions

To develop and enhance CGA, the authors in [39] have proposed an approach that detects the processors automatically on a machine and generates the number of equivalent working threads to calculate the Hash2 condition. In all the cores, the computation of CGA is implemented by assigning a parallel mechanism. In general, when one condition of CGA Hash2 thread is satisfied, the other will stop. In the identical approach, the speedup time will also increase if the number of cores in the computing devices increases. In [40], the CGA mechanism proposed requires less than 10 modular multiplications. This mechanism accomplishes pickup executions through two steps. (1) It selects the productive signature scheme with little variation in the Feige–Fiat–Shamir scheme. (2) Then, the crypto parameters scheme of the signature are tuned to secure the CGA quality. The only concern in this approach is that it calls for additional fitting assets (processors), which influence and restrict the abilities of devices, node versatility, and adversely.

The enhanced version of the protocol of CGA is known as CGA++ [41]. This protocol enhances the general security, and many attacks related to CGA are disposed. Initially, the alteration is made by considering the subnet prefixes of Hash2 calculations that could avoid the attacks. The verifier not only checks the link-local addresses but also verifies full IPv6 addresses. In [42], the utilization expansions and upgrades are also suggested to verify CGA in the annihilation of the DAD algorithm against the DoS attack. The TS used is additionally proposed as CGA inside option when it runs single. In addition, it is not considered a part of SEND. This CGA

is affected more by privacy-related attacks and can be resolved by tagging a CGA address for a lifetime. In this case, a tradeoff in the method occurs, and this arrangement between privacy and security is practical. In [43], the standard CGA is suggested with few alterations. It is proposed because it is mostly used to generate CGA. The running time of the upper bound of CGA is taken as input in the adjusted CGA generated algorithm, and revoking brute force is resolved by the yield of the sec value. The running time of the Hash2 value is changed in this altered CGA. This proposed algorithm is called Time-based CGA (TB-CGA).

Enhanced CGA [44] is presented as Elliptic Curve Cryptography (ECC) and Elliptic Curve DSA (ECDSA) in which RSA is first supplemented. Later on, it uses Graphical Processing Units (GPGPU) with general-purpose calculations. The alterations in the CGA generation method thus provide permission to the connected node that recently joined the link to generate the CGA address rapidly [45]. The processing is included in the CGA method ahead of time to perform the key-pair server node operation. The generation time decreases gradually as lengthy and time-consuming computation is performed on a server. This proposed method shows better performance. However, it relies more on an external server. If different server nodes are attacked, then the new nodes will not be able to join the network. To improve the computational speed of CGA, a parallelized CGA generation process is used with available resources in a trusted server [46]. It is also focused on malicious nodes on overload that influence the existing network. Here, trusted management is used, which is capable of finding and isolating the malicious nodes to remove possible incentive malicious behavior. Tab. 3 listed below summarizes the authors' work to improve CGA.

Table 3: CGA countermeasures summary

Solution	Strengths	Weaknesses
[39]	Speedup generation time.	Parallelization needs more processors
[40]	More efficient and lighter than original CGA	Call for more processors
[41]	Have higher security compare to standard CGA	More cost, as it incorporates subnet prefix in Hash2 calculation
[42]	The time needed is significantly decreased	Tradeoffs between privacy and security.
	Generate the key pair for the CGA algorithm on-the-fly enhance the security and protect privacy	More work and arrangements needed to practice it.
		Platform limited.
[43]	CGA is the vulnerability of attacks	It involves tradeoffs between privacy and security
[44]	Reduce the generation cost of CGA	Re-design the structure of CGA from standards
[45]	Reduced CGA generation cost	Rely on the outer server.
		Single point of failure to joined and new nodes.
[46]	Detect and isolate malicious nodes efficiently.	Software management system based

4 Methodology and Results

This section describes the steps and methodology to derive two models that tradeoff between SEND-based CGA security and performance. The two models are the CGA-Lighter and Locked-CGA models. Two main computational programs are used to develop the two models, namely, Open Secure Sockets Layer (OpenSSL) and Waikato Environment for Knowledge Analysis (Weka). OpenSSL is utilized to implement the CGA-Lighter model and reduce CGA generation costs using a light hash function, namely, MD5. Weka is used to implement the Locked-CGA model, including various components and functions, which are explained in the succeeding section. Moreover, components of the Locked-CGA model, such as the monitor, processor, and response controller, are likewise explained. The operation stages and workflow of the Locked-CGA model during its lifespan are also illustrated. The operation stages include the CGA traffic monitoring stage, collection stage, processing stage, and response control stage. Details on how the CGA-Lighter model can improve CGA performance are presented below.

4.1 CGA-Lighter Model

This model aims to minimize CGA generation costs to address the deficiency of the SEND protocol. MD-5 is a message digest-designed algorithm that takes an arbitrary length message as input to output a 128-bit input message digest or fingerprint. Yielding two messages with a similar message digest or generating a message that entails a predetermined message digest may be impossible. In 32-bit machines, the MD-5 algorithm is well designed to work efficiently and rapidly. Furthermore, the MD-5 algorithm can be coded compactly, as it does not require any extra substitution tables. According to the literature [47,48], the MD-5 hash function is fast and consumes less time to hash a construct.

Replacing the SHA-1 hash function with the MD-5 hash function in CGA generation can help reduce costs [49]. Although SHA-1 is more secure than MD-5, the latter should be considered. For instance, user mobility and mobile data demands increased recently [50–52]. In a mobile environment, when nodes have limited resources, using a heavy hash function to generate cryptographic addresses will limit the performance of a network and affect it negatively. The MD-5 and SHA-1 algorithms are considered secure, as no known methods have the ability to locate collisions except with brute force, which requires many years of breakthrough for one big message digest.

Although SHA-1 is more secure than MD-5, computing a message digest with SHA-1 costs more. In terms of security issues, SHA-1 is ideal. However, when speed is the primary concern, MD-5 is ideal and adequately secure in multiple applications [48]. CGA sequence generation is conducted normally in this proposed method except for the hash function construction using MD-5.

OpenSSL is used to implement the CGA-Lighter model, which is a software library used in applications that require secure communications over computer networks against eavesdropping or need to ascertain the identity of a party at the other end. OpenSSL is used extensively in Internet web servers and the majority of websites. OpenSSL contains SSL open-source implementation and transport layer security (TLS) protocols. The core library, which is written in the C programming language, implements essential cryptographic functions and provides various utility functions. Wrappers allowing the use of the OpenSSL library in various computer languages are available. Versions are available for most Unix and Unix-like operating systems (including Solaris, Linux, macOS, QNX, and various open-source BSD operating systems); OpenVMS; and Microsoft Windows [53].

4.2 Testing Scenario

In practice, a standard CGA considers a single hash value, which computes Hash1 and Hash2 in CGA specifications as two independent one-way hash values. The second Hash2 extension increases brute-force attack costs but reflects an increase in the hash output value length required for an IPv6 address, which is written into the interface ID portion. The sec value depends mostly on the Hash2 computational complexity. The address generated against the brute-force attack is used to indicate the security level of the sec using unsigned 3-bit integer values between 0 and 7 (0 being the least secure and 7 being the most secure).

Scenarios for three different CGA sec parameter values (i.e., sec = 1, sec = 2, and sec = 3) are implemented with five different computer specifications and processor speeds. The average of five runs with different computer specifications is collected and compared with that of a standard CGA. The scenarios are conducted for three different CGA sec security scale values with processor speeds ranging from 2.0 GHz to 3.2 GHz. Three charts for sec values 0, 1, and 2 for five rounds are presented below. In general, the three different outputs for sec values 0, 1, and 2 using five different processors demonstrate a fast generation time when the CGA-lighter model is used. Consequently, computer resources and processor cycles are saved. A comparison between the standard CGA and CGA-Lighter results is shown when the sec value is set to 0 (Fig. 2), 1 (Fig. 3), and 2 (Fig. 4), as illustrated below.

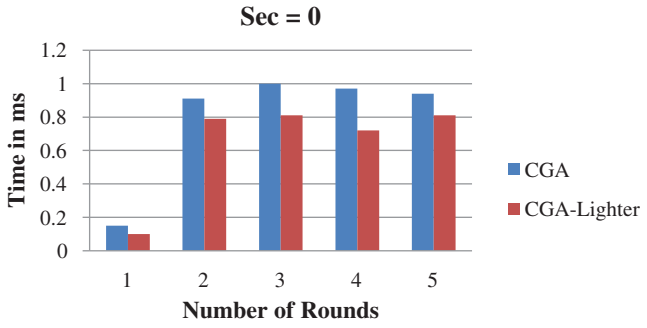

Figure 2: Sec variable with 0

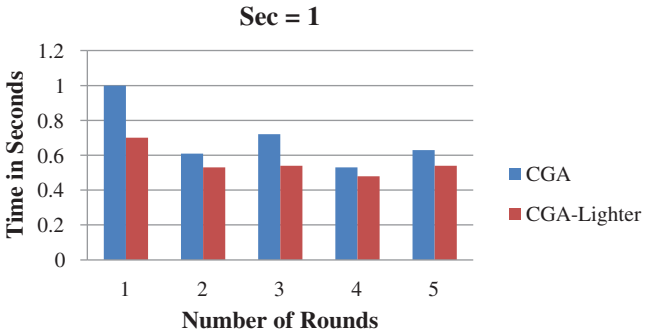

Figure 3: Sec variable with 1

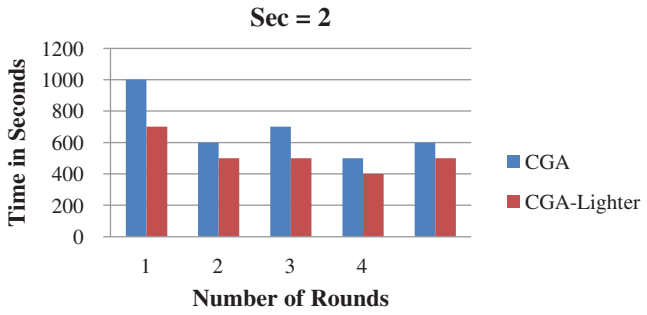

Figure 4: Sec variable with 2

4.3 Locked-CGA Model

The Locked-CGA model is designed to deal with the CGA DoS attacks described below.

4.3.1 DoS Attack Against the DAD CGA

According to RFC 3756, IPv6 is highly vulnerable to DoS attacks in the DAD algorithm [54,55]. A victim node employs the DAD algorithm with a temporary address, and a reply from an attacker saying that the address is in use is received by the node. This process is how a victim node is prevented from joining and configuring the IP address of a network.

4.3.2 DoS Attack Against the CGA Parameters

In such attacks, CGA vulnerability is generally high, as replies of signed messages are sniffed out and stored in a victim's node for a later reply. A sender's CGA parameters are used by an attacker to execute a DoS attack by replying or resending a host-enabled CGA. To verify the sender interface identifier (IID), Hash1 must calculate the CGA-enabled receiver. This verification process enables a sender to send the parameters to a receiver. Hash1 fails when the parameters are modified. This mechanism is employed between a CGA-enabled sender and receiver to prevent communication from an attacker.

The Locked-CGA model is developed by utilizing an artificial neural network (ANN) and backpropagation algorithm in the Weka 3.8 suite. Fig. 5 describes the main components of the Locked-CGA model, and the functions of the model are described in Tab. 4. The data structure of the Locked-CGA model is explained in Tab. 5.

Weka is used to implement this model, which is a machine learning software suite written in Java developed at the University of Waikato, New Zealand. Weka contains a collection of visualization tools and algorithms for data analysis and predictive modeling, with graphical user interfaces for easy access. Two sets, that is, a training set and test set, are used for the multilayer perceptron (MLP) neural network training in Weka.

a. Training set: A set of examples used for learning that is to fit the parameters [i.e., weights] of the classifier.
b. Test set: A set of examples used only to assess the performance [generalization] of a fully-specified classifier.

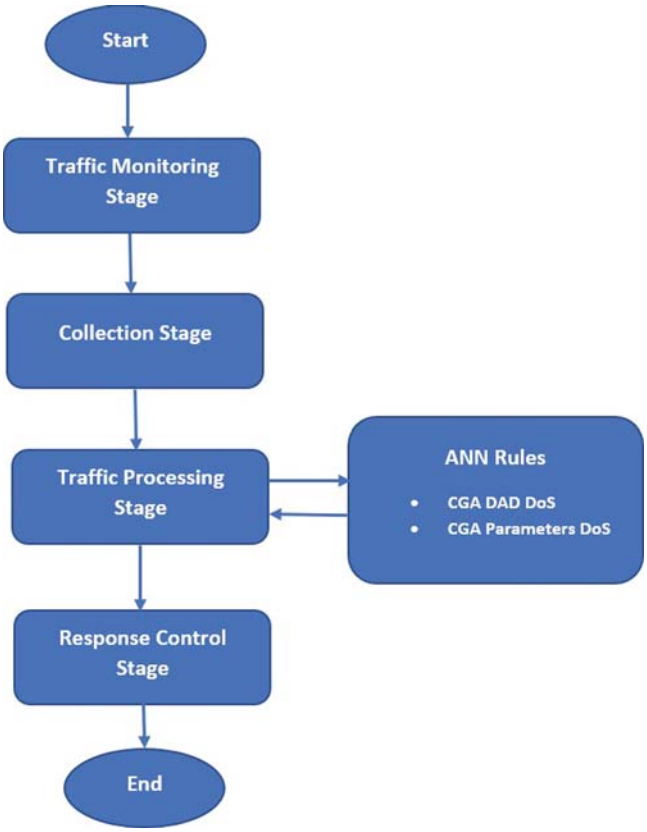

Figure 5: Locked-CGA components

Table 4: Functions of locked-CGA components

Locked-CGA model stage	Function
Monitor	Monitor messages between two CGA parties. Monitor the CGA verification timer.
Collector	Collect frames TS and sender IID.
Processor	Matching collected information against rules.
Response controller	Act to the CGA communicated party according to rules.

Table 5: Locked-CGA data structure

Data	Notation	Length
Subnet prefix	SP	64-bit
Interface identifier	IID	64-bit
Public-key	K_pub	Variable-length
Modifier	m	128-bit
Collision count	CC	8-bit
Time stamp	TS	Variable-length
Counter	c	3-bit

Back Propagation algorithm is used for the network training. The initialization of all weights to small random numbers is done until satisfied DO conditions [56] in this algorithm occurred. For each training example DO

a. Input the training example to the network and compute the training outputs
b. For each output unit k

$$\partial k \leftarrow ok\,(1-ok)\,(tk-ok)$$

c. For each hidden unit h

$$\partial h \leftarrow oh\,(1-oh)\sum k \in outputs\, wh,k\partial k$$

d. Update each network weight

$$wi,j \leftarrow wi,j+\Delta wi-j$$

4.3.3 Testing Scenarios

For the Locked-CGA model, two testing scenarios, namely, C1 (CGA parameter DoS attack) and C2 (CGA DAD DoS attack), are implemented using Weka. For scenario C1, a CGA parameter DoS attack is executed between CGA parties during the CGA verification procedure. The recorded performance metric for this scenario is the verification procedure time. For scenario C2, a CGA DAD DoS attack is executed between CGA parties' verifier and neighbor nodes. Attacker detection is based on two parameters, that is, the frame TS and IID. The controller of the Locked-CGA model takes a decision based on the gained values of this parameter and compares it with existing ANN rules. Tab. 5 presents the notations for the Locked-CGA model along with the meaning and length.

Figs. 6 and 7 show the pseudocode of the algorithms used to defend against the CGA DAD DoS attack and CGA parameter DoS attack, respectively. For the Locked-CGA pseudocode, four variables, namely, the IID, TS, network synchronized time (T), and IID table (IIDt), are traced during the verification stage of the CGA algorithm. The algorithm checks whenever a SEND-based NA message arrives from a neighbor claiming that the solicited CGA address is in use during the DAD procedure. If the time difference between the TS and T of that packet is less than five seconds and the address of the NA IID exists in the IIDt, then the sender of that NA message will be classified as an attacker, and the CGA verifier will configure the claimed address as its own. The IPv6 self-configuration address includes the newly joined SEND-based node in the DAD procedure, which typically takes at least three to four seconds, thereby indicating that it is being investigated [57] by the testbeds. This investigation is why five seconds is chosen as the threshold. If the aforementioned conditions are not met, then only the DAD counter CC of the CGA verification will be completed, as it is delayed in the last part of the CGA verification algorithm rather than at the beginning, similar to a standard algorithm.

The most common IID performance metrics are false positive rates and detection accuracy rates. A low percentage of false positive and false negative rates indicate that the detection mechanism is accurate and trusted. To evaluate and verify the CGA-Locked model, we select a detection accuracy ratio metric that represents the percentage of success in detecting the aforementioned CGA DoS attack.

```
1  Procedure VerifyCGA (m, CC, SP, K_pub, IID, TS, T, IIDt)
2  Concat := concatenate (m,SP,CC,K_pub)
3  Digest := SHA1(Concat)
4  Hash1:= Digest [0:8]
5  Hash1[0] := Hash1[0]
6  IID:= CGA [8:16]
7  IID [0] := IID [0]
8  IF Hash1 ≠ IID
9  Return False
10 End IF
11 IF T − TS ≤ 5
12 IID = IIDt
13 Return False
14 End IF
15 IF CC > 2 or CGA [0:8] ≠ SP
16 Return False
17 End IF
```

Figure 6: Locked-CGA algorithm (CGA DAD DoS attack)

```
1  Procedure VerifyCGAProcedure VerifyCGA (m, CC, SP, K_pub, IID, c)
2  IF CC > 2 or CGA [0:8] ≠ SP Return False End IF
3  IF CC > 2 or CGA [0:8] ≠ SP
4  Return False
5  End IF
6  While c <= 2 do
7   Concat := concatenate (m,SP,CC,K_pub)
8  Digest := SHA1(Concat)
9   Hash1:= Digest [0:8]
10 Hash1[0] := Hash1[0]
11 IID:= CGA [8:16]
12 IID [0] := IID [0]
13 End
14 IF Hash1 ≠ IID
15 Return False
16 End IF
```

Figure 7: Locked-CGA algorithm (CGA parameters DoS attack)

A counter variable (c) is traced whenever the CGA verification algorithm begins between two CGA parties. A two-second difference threshold is chosen, because the CGA verification time when the RSA key length is 1024 and the scaling factor sec is 0 or 1 is less than one second in a moderate Pentium processor with a speed of 2.4 Mh [58]. Moreover, two seconds is a reasonable threshold, considering limited processor speeds and other RSA key lengths.

The results of scenarios C1 and C2 are compared with the outcome of the standard CGA verification algorithm under the same attacks. The Locked-CGA model demonstrates superior security when implemented. The results of the C1 scenario under the CGA DAD DoS attack are shown in Fig. 8. The figure shows that after a five-second running attack, the DoS attack is identified, and the CGA verification stops. In the C2 scenario, the Locked-GA model sufficiently manages to eliminate the verification parameter DoS attack, as shown in Fig. 9. Once an attacker

reaches the nonlegitimate node behavior threshold, the Locked-CGA model ignores the DAD replies from a specific IID.

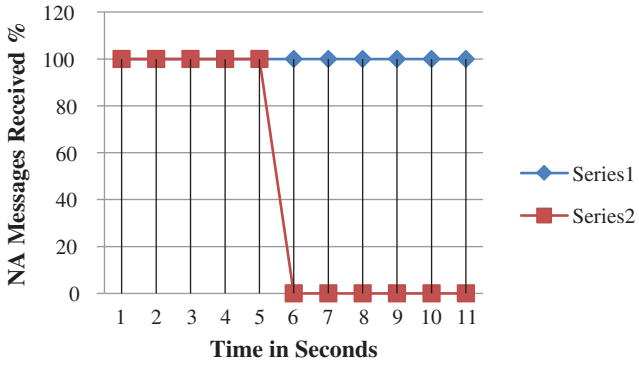

Figure 8: Results of C1 scenario

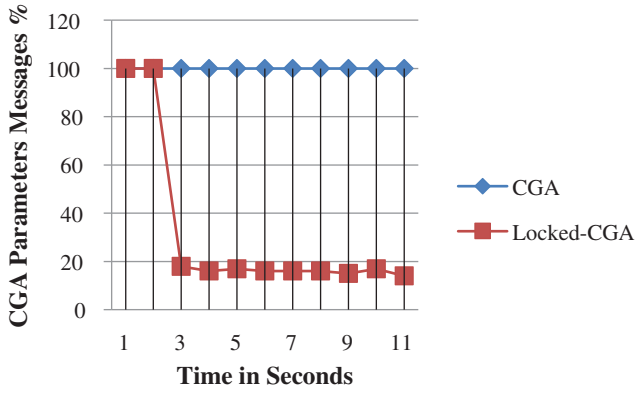

Figure 9: Results of C2 scenario

4.4 Summary

The proposed CGA-Lighter and Locked-CGA model have increased the performance of IPv6 protocol security issues. Both models have the potential to balance between the security and performance of SEND-based CGA successfully for the following reasons. First, CGA parameters respond to DoS attacks, and CGA-DAD DoS attacks have become detectable and easy to recognize. Second, the primary drawback of SEND-based CGA has been solved; it became lighter and more compatible for constrained devices with limited resources and specifications. Third, a security metric, such as the detection accuracy rate, is examined in this research to check the efficiency of the second model (Locked-CGA) to detect SEND-based CGA DoS attacks.

5 Conclusion

In this study, we proposed two new CGA models, namely, CGA-Lighter and Locked-CGA. Given the security issues in the present model, this new model was projected to balance the security and performance of SEND-based CGA and IPv6 NDP. The generation times (computation complexity) of CGA for three different sec values were investigated to verify the efficiency of

the first proposed model (CGA- Lighter). In these testing scenarios, the model was tested in five runs with five different machine specifications. For network security, the performance metric was tested in two scenarios, namely, for verification time and performance metrics. These metrics were selected because they are affected by the operations of the IPv6 network. We conclude that the performance of CGA is improved. Moreover, the consumption of resources is reduced, and CGA is more secured from the time against DoS attacks, as it uses packet TS and senders IID.

6 Limitations and Future Works

The Locked-CGA is limited by a legitimated node that may not cut off the threshold and become blocked. This limitation increases the possibility of obtaining false-positive rates. According to DAD internal mechanism and programming, this situation rarely occurs because once the node has passed the DAD check, it will not change the obtained address until it is rebooted again. Furthermore, the addition of TS in the CGA verification may increase network bandwidth consumption. However, this amount of consumption is also not considered a serious drawback, as verification is not frequently performed within a communication link and the bandwidth within a local link is always not an issue.

The need for more IP addresses has increased with the growth of connected devices to the Internet and the prevalence of the Internet of Things (IoT) [59,60]. Moreover Mobile *Ad hoc* Networks (MANETs) necessitate the deployment of IPv6 [61,62]. Therefore, further work is needed to address the rest of the CGA attacks and safely migrate the Internet to a native IPv6 infrastructure. The proposed work has demonstrated a new method to defend two attacks of CGA. However, subsequent attacks have not been covered. Therefore, this subject could be addressed in future works.

a. Global Time-Memory Trade-off Attack.
b. Alternate key pair Hashing is discovered of the victim's Node Address.
c. A private key identification of the victim node.

Acknowledgement: The authors would like to acknowledge the support of Network Communication Technology (NCT) Research Groups, FTSM, UKM in providing facilities for this research. This paper is supported under the Dana Impak Perdana UKM DIP-2018-040 and Fundamental Research Grant Scheme FRGS/1/2018/TK04/UKM/02/7.

Funding Statement: This work is supported by Dana Impak Perdana fund, no. UKM DIP-2018-040 and Fundamental Research Grant Scheme fund no FRGS/1/2018/TK04/UKM/02/7 under Author R. Hassan.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Zulkiflee, M. Azmi, S. Ahmad, S. Sahib and M. Ghani, "A framework of features selection for IPv6 network attacks detection," *WSEAS T Transactions on Communications*, vol. 14, no. 46, pp. 399–408, 2015.

[2] A. S. Ahmed, R. Hassan and N. E. Othman, "Security threats for IPv6 transition strategies: A review," in *Proc. ICE2T*, Kuala Lumpur, Malaysia, pp. 83–88, 2014.

[3] A. Al-Ani, M. Anbar, I. H. Hasbullah, R. Abdullah and A. K. Al-Ani, "Authentication and privacy approach for DHCPv6," *IEEE Access*, vol. 7, pp. 73144–73156, 2019.

[4] Supriyanto, I. H. Hasbullah, R. K. Murugesan and S. Ramadass, "Survey of internet protocol version 6 link local communication security vulnerability and mitigation methods," *IETE Technical Review*, vol. 30, no. 1, pp. 64–71, 2013.

[5] S. Hagen, *IPv6 Essentials*, 3rd ed., vol. 1. Newton, Massachusetts, United States: O'Reilly Media Incorporation, 2006.

[6] J. L. Shah and H. F. Bhat, "Towards a secure IPv6 autoconfiguration," *Information Security Journal: A Global Perspective*, vol. 29, no. 1, pp. 14–29, 2020.

[7] S. Deering and R. Hinden, *Internet Protocol, Version 6 (IPv6) Specification*. California, United States: Internet Engineering Task Force, 1998.

[8] J. Davies, *Understanding IPv6*, 2nd ed., vol. 1. Washington, United States: Microsoft, 2003.

[9] M. Blanchet, *Migrating to IPv6: A Practical Guide to Implementing IPv6 in Mobile and Fixed Networks*, 1st ed., vol. 1. New Jersey, United States: John Wiley and Sons, 2009.

[10] K. Batiha, K. Batiha and A. AbuAli, "The need for IPv6," *International Journal of Academic Research*, vol. 3, no. 3, pp. 431–448, 2011.

[11] A. S. Ahmed, R. Hassan and Z. M. Ali, "Eliminate spoofing threat in IPv6 tunnel," in *Proc. ICIDT*, Jeju, South Korean, pp. 218–222, 2012.

[12] A. Alsa'deh and C. Meinel, "Secure neighbor discovery: Review, challenges, perspectives, and recommendations," *IEEE Security & Privacy*, vol. 10, no. 4, pp. 26–34, 2012.

[13] M. Anbar, R. Abdullah, R. M. Saad, E. Alomari and S. Alsaleem, "Review of security vulnerabilities in the IPv6 neighbor discovery protocol," in *Proc. ICISA*, Singapore, pp. 603–612, 2016.

[14] A. El Ksimi and C. Leghris, "Towards a new algorithm to optimize IPv6 neighbor discovery security for small objects networks," *Security and Communication Networks*, vol. 2018, no. n6, pp. 1–11, 2018.

[15] E. Mahmood, A. H. Adhab and A. K. Al-Ani, "Review paper on neighbour discovery protocol in IPv6 link-local network," *International Journal of Services Operations and Informatics*, vol. 10, no. 1, pp. 65–78, 2016.

[16] A. El Ksimi and C. Leghris, "An enhancement approach for securing neighbor discovery in IPv6 networks," in *Proc. ICMSPN*, Cham, pp. 54–69, 2018.

[17] A. Alsadhan, A. Hussain, P. Liatsis, M. Alani, H. Tawfik *et al.,* "Locally weighted classifiers for detection of neighbor discovery protocol distributed denial-of-service and replayed attacks," *Transactions on Emerging Telecommunications Technologies*, vol. 13, no. 12, pp. 175, 2019.

[18] G. Song, H. Wang and F. Liu, "Using fdad to prevent dad attack in secure neighbor discovery protocol," *Security and Communication Networks*, vol. 2020, pp. 1–15, 2020.

[19] A. K. Al-Ani, M. Anbar, S. Manickam, C. Y. Wey, Y. B. Leau *et al.,* "Detection and defense mechanisms on duplicate address detection process in IPv6 link-local network: A survey on limitations and requirements," *Arabian Journal for Science and Engineering*, vol. 44, no. 4, pp. 3745–3763, 2019.

[20] A. K. Al-Ani, M. Anbar, S. Manickam, A. Al-Ani and Y. B. Leau, "Proposed dad-match mechanism for securing duplicate address detection process in IPv6 link-local network based on symmetric-key algorithm," in *Proc. ICCST*, Singapore, pp. 108–118, 2017.

[21] A. R. Choudhary and A. Sekelsky, "Securing IPv6 network infrastructure: A new security model," in *Proc. ICTHS*, Waltham, Massachusetts, United States, pp. 500–506, 2010.

[22] A. AlSa'deh, H. Rafiee and C. Meinel, "Secure neighbor discovery: A cryptographic solution for securing IPv6 local link operations," in *Theory and Practice of Cryptography Solutions for Secure Information Systems*, Pennsylvania, United States: IGI Global, pp. 178–198, 2013.

[23] X. Yang, T. Ma and Y. Shi, "Typical dos/ddos threats under IPv6," in *Proc. ICCGI*, Guadeloupe City, Guadeloupe, pp. 55, 2007.

[24] O. E. Elejla, M. Anbar and B. Belaton, "Icmpv6-based dos ad ddos attacks and defense mechanisms," *IETE Technical Review*, vol. 34, no. 4, pp. 390–407, 2017.

[25] F. A. Barbhuiya, G. Bansal, N. Kumar, S. Biswas and S. Nandi, "Detection of neighbor discovery protocol-based attacks in IPv6 network," *Networking Science*, vol. 2, no. 3, pp. 91–113, 2013.

[26] J. L. Shah and J. Parvez, "IPv6 cryptographically generated address: Analysis and optimization," in *Proc. ICAICTC*, Bikaner India, pp. 1–6, 2016.

[27] J. W. Bos, O. Özen and J. P. Hubaux, "Analysis and optimization of cryptographically generated addresses," in *Proc. ICIS*, Berlin, Heidelberg, Germany, pp. 17–32, 2009.

[28] G. O'shea and M. Roe, "Child-proof authentication for mIPv6 (CAM)," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 2, pp. 4–8, 2001.

[29] P. A. Nikander, *Scaleable Architecture for IPv6 Address Ownership*. California, United States: Internet Engineering Task Force, 2001.

[30] G. Montenegro and C. Castelluccia, "Statistically unique and cryptographically verifiable (SUCV) identifiers and addresses," in *9th Annual Network and Distributed System Security Symp.*, San Diego, California, USA, 2002.

[31] T. Aura, "Cryptographically generated addresses (CGA)," in *Proc. ICIS*, Berlin, Heidelberg, Germany, pp. 29–43, 2003.

[32] J. H. Park, K. H. Choi, J. S. Kim, C. I. Cho, H. J. Jang *et al.,* "A survey of the secure neighbor discovery (send) and multi-key cryptographically generated addresses (MCGAs)," in *Proc. ICACT*, Okamoto, Kobe, Japan, pp. 2124–2127, 2007.

[33] A. S. A. M. S. Ahmed, R. Hassan and N. E. Othman, "IPv6 neighbor discovery protocol specifications, threats and countermeasures: A survey," *IEEE Access*, vol. 5, pp. 18187–18210, 2017.

[34] R. Hassan, A. S. Ahmed and N. E. Osman, "Enhancing security for IPv6 neighbor discovery protocol using cryptography," *American Journal of Applied Sciences*, vol. 11, no. 9, pp. 1472–1479, 2014.

[35] S. Qadir and M. U. Siddiqi, "Cryptographically generated addresses (CGAS): A survey and an analysis of performance for use in mobile environment," *International Journal of Computer Science and Network Security*, vol. 11, no. 2, pp. 24–31, 2011.

[36] J. Arkko, J. Kempf, B. Zill and P. Nikander, *Secure Neighbor Discovery (SEND)*. California, United States: Internet Engineering Task Force, 2005.

[37] A. Alsadeh, H. Rafiee and C. Meinel, "Cryptographically generated addresses (CGAS): Possible attacks and proposed mitigation approaches," in *Proc. ICCIT*, Chengdu, China, pp. 332–339, 2012.

[38] V. Vasić, A. Kukec and M. Mikuc, "Deploying new hash algorithms in secure neighbor discovery," in *Proc. IC STCN*, Split, Croatia, pp. 1–5, 2011.

[39] A. Alsa'deh, H. Rafiee and C. Meinel, "Stopping time condition for practical IPv6 cryptographically generated addresses," in *Proc. ICIN*, Bali, Indonesia, pp. 257–262, 2012.

[40] C. Castelluccia, "Cryptographically generated addresses for constrained devices," *Wireless Personal Communications*, vol. 29, no. 3, pp. 221–232, 2004.

[41] J. W. Bos, O. Özen and J. P. Hubaux, "Analysis and optimization of cryptographically generated addresses," in *Proc. ICIS*, Berlin, Heidelber, Germany, pp. 17–32, 2009.

[42] H. Rafiee, A. Alsa'deh and C. Meinel, "Multicore-based auto-scaling secure neighbor discovery for windows operating systems," in *Proc. ICIN*, Bali, Indonesia, pp. 269–274, 2012.

[43] A. lsadeh, H. Rafiee and C. Meinel, "Cryptographically Generated Addresses (CGAS): Possible attacks and proposed mitigation approaches," in *Proc. ICCIT*, Chengdu, China, pp. 332–339, 2012.

[44] T. Cheneau, A. Boudguiga and M. Laurent, "Significantly improved performances of the cryptographically generated addresses thanks to ecc and gpgpu," *Computers & Security*, vol. 29, no. 4, pp. 419–431, 2012.

[45] S. Guangxue, W. Wendong, G. Xiangyang, Q. Xirong, J. Sheng *et al.,* "A quick cga generation method," in *Proc. ICFCC*, Wuhan, China, pp. V1–V769, 2010.

[46] M. Moslehpour and S. Khorsandi, "Improving cryptographically generated address algorithm in IPv6 secure neighbor discovery protocol through trust management," *International Journal of Humanities and Social Sciences*, vol. 10, no. 6, pp. 1010–1014, 2016.

[47] M. Blaze, "Efficient, dos-resistant, secure key exchange for Internet protocols," in *Proc. IWSP*, Berlin, Heidelberg, Germany, pp. 40–48, 2001.

[48] W. Aiello, S. M. Bellovin, M. Blaze, J. Ioannidis, O. Reingold *et al.,* "Efficient, dos-resistant, secure key exchange for Internet protocols," in *Proc. ACM CCCS*, Washington, DC, USA, pp. 48–58, 2002.

[49] A. S. Ahmed, N. H. A. Ismail, R. Hassan and N. E. Othman, "Balancing performance and security for IPv6 neighbor discovery," *International Journal of Applied Engineering Research*, vol. 10, no. 19, pp. 40191–40196, 2015.

[50] A. Alsa'deh and C. Meinel, "Secure neighbor discovery: Review, challenges, perspectives, and recommendations IEEE," *Security & Privacy*, vol. 10, no. 4, pp. 26–34, 2012.

[51] B. Park, "Threats and security analysis for enhanced secure neighbor discovery protocol (send) of IPv6 ndp security," *International Journal of Control and Automation*, vol. 4, no. 4, pp. 179–184, 2011.

[52] H. I. Zawia, R. Hassan and D. P. Dahnil, "A survey of medium access mechanisms for providing robust audio video streaming in IEEE 802.11 aa standard," *IEEE Access*, vol. 6, pp. 27690–27705, 2011.

[53] J. Viega, M. Messier and P. Chandra, *Network Security with Openssl: Cryptography for Secure Communications*, 1st ed., vol. 1. Newton, Massachusetts, United States: O'Reilly Media Incorporation, 2002.

[54] S. Guangjia, W. Hui and W. Hangjun, "Using multi-address generation and duplicate address detection to prevent DoS in IPv6," *IET Communications*, vol. 13, no. 10, pp. 1390–1396, 2019.

[55] A. K. Al-Ani, M. Anbar, A. Al-Ani and D. R. Ibrahim, "Match-prevention technique against denial-of-service Attack on Address Resolution and Duplicate Address Detection Processes in IPv6 Link-Local Network," *IEEE Access*, vol. 8, pp. 27122–27138, 2020.

[56] R. Rojas, *The Backpropagation Algorithm*. Berlin, Heidelberg, Germany: Springer, 1996.

[57] A. S. Ahmed, R. Hassan, E. Othman, I. Ahmad and Y. Kenish, "Impacts evaluation of dos attacks over IPv6 neighbor discovery protocol," *Journal of Computer Science*, vol. 15, no. 5, pp. 702–727, 2019.

[58] T. Cheneau, A. Boudguiga and M. Laurent, "Significantly improved performances of the cryptographically generated addresses thanks to ECC and GPGPU," *Computers & Security*, vol. 29, no. 4, pp. 419–431, 2010.

[59] M. Zakri and R. Hassan, "The implementation of internet of things using test bed in the ukmnet environment," *Asia Pacific Journal of Information Technology and Multimedia*, vol. 8, no. 2, pp. 1–17, 2019.

[60] P. I. R. Grammatikis, P. G. Sarigiannidis and I. D. Moscholios, "Securing the nternet of things: Challenges, threats and solutions," *Internet of Things*, vol. 5, no. 7, pp. 41–70, 2019.

[61] Z. Ismail, R. Hassan, A. Patel and R. Razali, "A study of routing protocol for topology configuration management in mobile ad hoc network," in *Proc. ICEEI*, Selangor, Malaysia, pp. 412–417, 2009.

[62] Z. Ismail and R. Hassan, "Performance of aodv routing protocol in mobile ad hoc network," in *Int. Symp. on Information Technology*, Kuala Lumpur, Malaysia, pp. 1–5, 2010.