Chapter 5

Link Layer Security Analysis and Optimization

5.1 Introduction

Link Local communication is one of the dominant features of IPv6 networks.IPv6 nodes use Link Local communication to discover nodes on the link, to determine their link layer addresses, to find the presence of routers on the link, to detect any duplicate addresses and to maintain reachability information to other nodes (RFC 3971).To achieve Link Local communication, IPv6 nodes use Neighbor Discovery (ND) Protocol. The protocol also serves as fundamental core in mobile communications enabling multihop communication which is essential for route discovery and data forwarding.Additionally in Mobile IPv6, Neighbor Discovery Protocol plays a critical role by banishing the need for foreign agents and allowing the mobile nodes to join new foreign networks. Neighbor Discovery forms an innate of Internet Control Message Protocol and uses ICMPv6 message format(AlSa'deh&Meinel, 2012).It also operates with same protocol number i.e. 58 but has its own design. TheNeighbor Discovery Protocol has been drafted as RFC 4861.The protocol has no equivalent mechanism in IPv4, however resembles in some functionality with its predecessor. The Neighbor Discovery Protocol replaces IPv4 Address Resolution Protocol (ARP) and ICMPv4 Router Discovery and Redirect.

As IPv6 fortifies inherent support to Internet Protocol Security (IPSec) for end to end communication over the internet, however this security protocol does not scale up to the link local communication that uses NDP(Supriyanto et al, 2013). The security at the link layer provides an important role as the Internet being an open network is vulnerable to be exploited by attackers from both outside and inside the network. Also a number of security firewalls typically focus on blocking external threats from outside the network without caring about the threats originating from inside the network. Thus, analyzing and mitigating the threat and vulnerabilities in the local network is inherently important.

The Neighbor discovery protocol is vulnerable to critical attacks because it assumes that all nodes on the link trust each other(Baig&Adeniye, 2011). However this viewpoint does not hold good in other scenarios like in case of wireless networks, where in any node can join the link with minimal authentication and start the communication channel. As a result, malign users would carry impersonation of authorized nodes by faking NDP messages to foster attacks. To counter this, Secure Neighbor Discovery protocol (SEND) was developed and drafted inthe RFC

3971 to secure link local communication. SEND uses authentication mechanisms like Cryptographically Generated addresses (CGA), RSA Digital Signatures, Nonce, Timestamps and X.509 certificate exchange techniques to secure NDP.The SEND ensures message integrity checks, prevents address impersonation and replay attacks, and also ensures the legitimacy of routers on the link. Although SEND is considered to be robust technique for protecting IPv6 NDP, its use and practical deployment is somewhat difficult to achieve. The CGA part of SEND is computationally heavy and bandwidth gobbling. SEND also lacks mature implementations in modern operating systems. Moreover, SEND itself can be vulnerable to some attacks. This chapter introduces SEND functionalities and messages. We also discuss the implementation and deployment challenges of IPv6 SEND. The chapter analyses the CGA security and computational complexity and finally proposes the techniques that can be used in optimizing the practical deployment of CGA in IPv6 networks.

5.2 Neighbor Discovery Protocol(NDP)

The Neighbor discovery protocol forms an inherent component of Internet Control Message Protocol for IPv6 (ICMPv6) and uses ICMPv6 message format as shown in figure 5.1.The structure of NDP messages consists of an ICMPv6 header, Neighbor Discovery message specific data and additional Neighbor discovery message options like Layer-2 MAC addresses, subnet prefixes, on link maximum transmission unit (MTU), redirect and router specific information(Hagen, 2006).

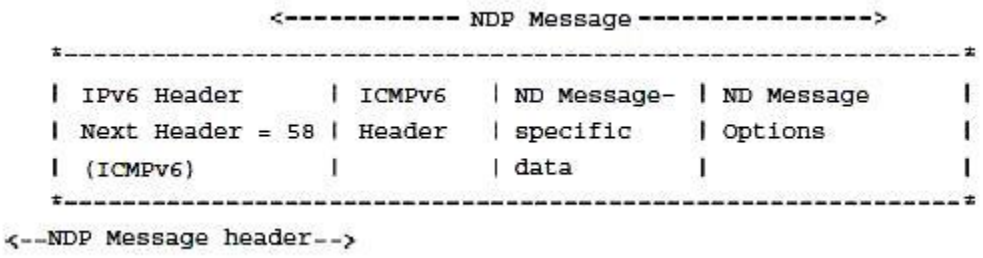

Figure 5.1 IPv6 NDP Format

RFC 4861 states the purpose of Neighbor Discovery as follows:

> *"The Neighbor Discovery Protocol is used by IPv6 nodes to discover each other's presence on the link, to ascertain each other's link-layer addresses, to search routers and to sustain reachability information about the route to active neighbors."*

To attain the aforementioned functionalities and services, NDP uses five different messages:

- *Router Solicitation (RS-Message type 133)*: Router Solicitation (RS) messages are directed by any node at any time to request all routers on the local link to promptly send Router Advertisement (RA) messages. RA messages are sent by routers periodically, but when a node first boots up, it normally sends an RS message in order to obtain the RA information such as prefixes and DNS server addresses instantly, instead of waiting for the next periodic transmission.
- *Router Advertisement (RA-Message type 134):* The Router Advertisement (RA) message broadcasts varied subnet-wide information to all nodes in a subnet (link). Every IPv6 router is capable of sending an RA message periodically (the time period is altered randomly to avoid all routers from sending simultaneously). Routers also respond with RA message in response to a Router Solicitation message from any internal node.
- *Neighbor Solicitation (NS-Message type 135):* Any IPv6 capable node can send a Neighbor Solicitation (NS) message to request a target node's link-layer address, while also advertising its own link-layer address to the destination node. NS messages are sent via multicast to the Solicited Node Multicast Address of the target node when the sending node is doing Address Resolution. They are sent via Unicast to the target node's link-local address when the sending node is doing Neighbor Unreachability Detection.
- *Neighbor Advertisement (NA-Message type 136):* Any IPv6 capable node must respond with a Neighbor Advertisement (NA) message in response to a Neighbor Solicitation (NS) message. A node can also volunteer to send an unsolicited Neighbor Advertisement message in order to disseminate new information quickly.
- *Redirect Message (RM-Message type 137):*The ND Redirect message is used to notify a host of a better first-hop node on the way to the intended destination node. The Redirect message can also be used to inform a host that the destination is in fact a neighbor (in its home subnet).

In addition to above, five different options are used that are inserted in the options field of a message. These options include Source Link Layer Address Option, Target Link Layer Address Option, Prefix information option, Redirected header option and MTU option.

5.3 Neighbor Discovery Security Issues

The Neighbor Discovery protocol being a link local protocol is susceptible to the following attacks.

- *Spoofing*

In this attack, an attacker node successfully impersonates other nodes address or interface identifier to leverage Man-in-the-Middle attacks (MITM).A malicious node alters packets from legitimate nodes to be redirected to some other link layer address. This can be achieved by sending spoofed source link layer address in a Neighbor Solicitation Message or sending spoofed Target Link layer address in a Neighbor Advertisement Message. If the spoofed address is a valid one, then as long as the malicious node responds to the Unicast Neighbor Solicitation messages sent as part of the Neighbor Unreachability Detection; packets will continue to be redirected.

- *Denial of Service(DoS)*

DoS are very common and efficacious attacks as it eats away substantial amount of system resources by generating fake requests which cannot be serviced. In ND protocol, malicious users can leverageDoS on node's Duplicate Address Detection (DAD) during an address auto configuration process to avert a node from obtaining legitimate address. A malicious node responds to every duplicate address detection message with a spoofed message of claiming a legitimate address. As a result, legitimate nodes are not able to configure their addresses.

- *Parameter Spoofing*

A Router Advertisement message contains many pieces of information used by end hosts while sending packets and a flag value to inform hosts whether to carry stateful address configuration. A malicious user could craft a valid Router Advertisement that duplicates the Router Advertisement from a legitimate default router. The attacker modifies the parameter values in such a way in order to disrupt legitimate traffic. For example; an attacker could broadcast false Router Advertisements specifying that the underlying network uses DHCP for address configuration which could cause other nodes to contact a nonexistent DHCP server and not get any publicly usable IP addresses at all.

- *Rouge Router*

In this type of attack, an attacker can falsely claim to be the default router on the link. As a result, all legitimate traffic gets redirected to the malicious user. It's very elementary for an attacker to configure itself as a default router on the link, but it's extremely difficult for a node to make distinction between spoofed and authorized Router Advertisements, especially for a newly connected node.

- *Replay*

In a replay attack (also known as playback attack) a valid data transmission is maliciously or fraudulently repeated or delayed. In this attack, an attacker obtains copy of the message sent by the legitimate user and at a later point of time tries to replay it.

In addition to the above mentioned problems, Neighbor Discovery (ND) Protocol also suffers from certain privacy issues(Choudhary, 2009). One such issue is related with the IPv6 Stateless Address Auto configuration (SLAAC).In IPv6 SLAAC, if Interface Identifier (IID) is generated on the basis of MAC address, it results in static IID which does not change over time and remains constant. Thus an attacker can associate the captured traffic from a certain IID to a specific device and track the node. Once the identity of the node gets tracked down, attacker can launch a range of attacks against the node. The first bold attempt to secure privacy of node was made in RFC 4941, "Privacy Extensions for Stateless Address Auto configuration in IPv6" by introducing the concept of changing the IID's over time.However; even this approach is not secure enough to protect against IP spoofing attacks.

5.4 Secure Neighbor Discovery (SEND) - A Cryptographic Solution

In order to address the security issues with Neighbor Discovery Protocol, authors (Arkko et al, 2002; RFC 3971, 2005)proposed the Secure Neighbor Discovery Protocol (SEND) to protect NDP messages. SEND is felicities in those scenarios in which physical security on the link is not assured (such as wireless networks).Since both the hosts as well as routers make use of NDP,it was therefore innate to develop a protocol that would secure communication at the link layer. The SEND is implemented as a security extension to the Neighbor Discovery protocol that enumerates a number of options such as Cryptographically Generated Addresses (CGA's), RSA Signature, Timestamp and Nonce. In addition to this,it also introduces two new ICMPv6

messages: certificate Path Solicitation (CPS) and Certificate Path Advertisement (CPA) for the router authorization process. The main building blocks of SEND protocol are:

- *Cryptographically Generated Address:*

SEND uses CGA, a cryptographic method for binding a public signature key to an IPv6 address(Qadir& Siddiqi, 2011). CGAs are used to ensure that the sender of a neighbor discovery message is the real "owner" of the claimed address. A public-private key pair is generated by all nodes before they can claim an address. A new NDP option, the CGA option, is used to carry the public key and associated parameters. CGA is formed by replacing the least-significant 64 bits of the 128-bit IPv6 address with the cryptographic hash of the address owner's public key. The messages are signed with the corresponding private key. Only if the source address and the public key are known can the verifier authenticate the message from that corresponding sender

- *RSA Signature:*

The RSA Signature option is used as an authentication mechanism to authenticate sender's identity. Every node before claiming an address must acquire a unique pair of public/private keys. Every message sent by the user is digitally signed by the private key that corresponds to the public key to be used in CGA generation algorithm. Thus signature works as a defense mechanism for spoofing CGA addresses.

- *TimeStamp:*

In order to certify protection against unsolicited advertisements (timely Router Advertisement and Redirect Message), SEND uses the timestamp option. The only assumption that is made is that all nodes must have their clocks synchronized in order to avert replay attacks.

- *Nonce:*

This option certifies that an advertisement is a fresh acknowledgement to a solicitation request sent earlier by the node.

- *Router Authorization(Authorization Delegation Discovery(ADD))*

  To validate and authenticate IPv6 routers to act as default gateways, SEND uses Authorization Delegation Discovery Mechanism .ADD uses the concept of electronic certificate that is issued by a trusted third party authority. A node before accepting a router as its default must configure itself with a trust anchor that can aid to certify the router via a

certification path. Thus node requires the router to provide its X.509 certificate path to a trust anchor which must be preconfigured on the node. A router that fails to provide the path to the trust anchor shouldn't be trusted.

5.5 IPv6 Cryptographically Generated Address (CGA)

Cryptographically Generated Address (CGA) are introduced as one of the dominant features of Secure Neighbor Discovery (SEND) Protocol used for IPv6 Security.CGA works without relying on any trusted third party authority or Public Key Infrastructure (PKI) and offers authentication mechanism for IPv6 addresses (Aura, 2005).This decentralized mechanism completely varies from the legacy approach wherein central authority issues electronic certificates to fortify the binding. In IPv6 CGA, the Interface Identifier's (IID) are formed by applying cryptographic hash function on the owner's public key and other parameters. With this technique, the IPv6 address is glued with the public key and thus helps address generator assert ownership of its address. This coupling can be corroborated by re-computation and comparison of interface identifier of sender and receiver. The notion of Cryptographically Generated Address was first realized by (O'shea& Roe, 2001) in child-proof authentication for Mobile IPv6 (MIPv6 CAM) protocol. They proposed a mechanism for authenticating location updates in Mobile IPv6. The idea was later refined by (Nikander, 2001).The authors (Montenegro &Castelluccia, 2002) in the paper titled "Statistically unique and cryptographically verifiable addresses" addressed the identifier ownership problem that hinders mechanisms like Binding Updates in Mobile IPv6.The RFC 3972 (Aura, 2005) defines the final model of CGA.The use of CGA has also been encouraged for authentication and mitigating Denial of Service (DoS) attacks in Mobile IPv6(Aura & Roe, 2006; Arkko, Haddad, & Vogt, 2007).

Though CGA is a proficient technique and offers considerable amount of security, it does possess some inherent limitations and performance bottlenecks. For generating a CGA, the sender selects the value of 3 bit Security parameter (sec) which ranges from 0 to 7 that determines the level of security of CGA against the brute force attack. On average, to satisfy the Hash-2 condition, the address generator tries $2^{16\times sec}$ brute force searches for a valid modifier (Aura, 2005). Similarly for address impersonation, the attacker tries $2^{16\times sec+59}$ brute force searches. Thus sec increases the computational cost for both address generator as well as attacker. The Sec value "0" and "1" are preferred because large sec values cause undesirable

amount of time for CGA calculation. For example, for sec value "2", the algorithm on a modern CPU takes hours to generate a valid modifier. This delay is significant and impacts performance on resource constrained devices like mobile phones and sensor nodes. The higher computation cost of CGA shadows its benefits and scruples its usage in IPv6.

CGA is also strongly bound to RSA algorithm for public/private key generation. Since higher RSA key size guarantees optimum security, it also introduces considerable delay in key pair generation which in turn affects the CGA generation time. Additionally it also increases the packet size of Neighbor Discovery messages impacting bandwidth of the channel.

Another important aspect of CGA computation is the Hash function.CGA uses SHA-1(160 bit hash) which undoubtedly is vulnerable to collision free attacks (Wang, Yiqun& Yu, 2005). The weaknesses in SHA-1 necessitate the need for finding alternate and stronger Hash functions (McDonald, Philip & Josef, 2009).Additionally authors in (Bos, Ozen& Hubaux,2009) have shown that global time-memory trade-off attack can be leveraged on CGA's at the cost of storage that vanquishes the effect of hash extension in the long run. Although assumed to be unrealistic in (Aura, 2005), the attack can be leveraged in future considering the exponential power of computing devices following Moore's Law. The author's also discuss about the lack of authentication in CGA and introduce digital signature scheme which unfortunately comes with additional cost.

The above mentioned drawbacks may preclude the use of CGA's leaving IPv6 networks susceptible to attacks and abuse.

5.5.1 CGA Specification

The fundamental use of CGA is to avert address impersonation/spoofing of existing and assigned IPv6 address in the network by an attacker (Aura, 2005). CGA's are also partially used for providing authentication with the help of public/private key pair, i.e. proving that the sender of packet is actually the owner of CGA. This is usually implemented with the help of digital signatures which can be verified by the receiver of the CGA. The CGA usage mandates that sender and receiver share some common parameters also known as CGA parameter data structure. This essentially comprises of the following (Aura, 2005):

- 128 bit randomly generated modifier
- 64 bit subnet prefix

- 8 bit collision count
- DER Encoded public key
- Extension fields(optional)

CGA also include another security parameter "sec" value which is a 3 bit unsigned integer determining the level of resistance against brute force search attack. The value of sec varies from 0 to 7 and is used as a hash extension technique that increases the hash value beyond the 64 bits. Thus the cost of creating a hash gets increased with no effect on its hash length (Aura, 2005).The CGA generation process is shown in figure 5.2.

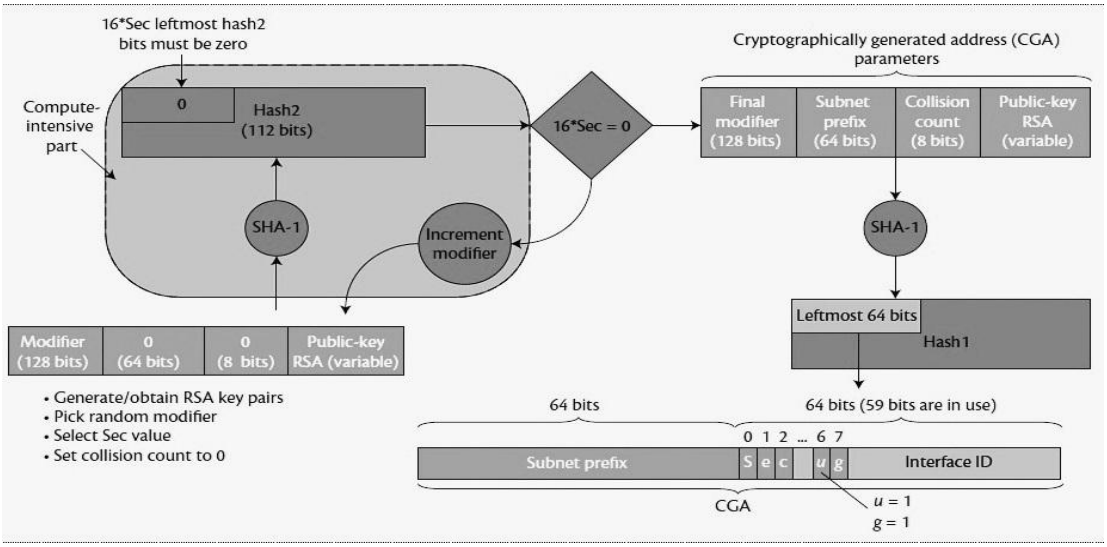

Figure 5.2 CGA Generation Algorithm

CGA generation algorithm starts by creating owner's public key and selecting the appropriate sec value. The owner needs to compute two different hash values (Hash1 and Hash2) using his public key and auxiliary parameters. First Hash2 value is calculated which is hash of concatenation of public key, modifier, a zero value of collision count and subnet prefix. Different values of Modifier are generated by the owner and the loop terminates when 16×sec leftmost bits of Hash2 equate to zero, i.e. if sec=1,the 16 leftmost bits of Hash2 must be zero.Hash2 computation is the most expensive part of CGA generation and is normally calculated once during CGA generation when new node joins the network. Once the final Modifier is obtained, it serves as an input for Hash1 calculation. The Hash1 value is obtained by hashing the concatenation of entire CGA data structure. The IID is derived by extracting the leftmost 64 bits of Hash1 and encoding the sec value into the three leftmost bits. The 7^{th} ("u") and 8^{th} ("g") bits

from the left are reserved. Finally, duplicate address detection is executed to ensure the uniqueness of address on the link.

CGA verification takes the CGA address and the CGA data structure as an input. To validate the binding between CGA address and public key, the Hash1 value is recomputed based on received CGA data structure and comparison is made with the IID of the sender. If the verification is successful, the receiver knows that the public key corresponds to address owner. The receiver can then validate signed messages from the sender by using the public key.

5.5.2 Cost Analysis of CGA

CGA's are computationally intensive and bandwidth heavy which preclude their usage in modern computer networks.CGA utilization increments the computational cost for both attacker as well as address generator. The address owner using brute force search needs O ($2^{16\times sec}$) hash function evaluations to find the right Modifier value and satisfy the Hash2 condition. Contrary to this, the attacker can impersonate a random node in a network, by finding an alternate public/private key pair. The Hash1 value thus obtained must match the IID of impersonated node. To do this, the attacker needs to perform (16×sec+59) hash function evaluations adding a cost of ($2^{16\times sec+59}$).

The attacker can satisfy the constraints on both Hash2 as well as Hash1.Starting with attack on Hash1, the attacker must perform $2^{59}\times T_1$hash function evaluations where T_1 is the time needed to compute Hash1. Once satisfied, the conditions on Hash2 for valid modifier needs to be fulfilled which takes $2^{16\times sec}$ hash function evaluations. Thus the total impersonation time when beginning with Hash1 equals $T_{H1} = (2^{59}\times T_1+T_2)\ 2^{16\times sec}$ where T_2 is the time needed to compute Hash2.

When beginning with Hash2, the attacker must perform $2^{16\times sec}\times T_2$hash function evaluations to satisfy condition on Hash2.Once fulfilled, Hash1 is verified at a cost of 2^{59}.Thus the total impersonation time when beginning with Hash2 becomes $T_{H2} = (2^{16\times sec}\times T_2+T_1)\ 2^{59}$.An attacker can select either of the two methods to curtail his attack vector. Thus,

$$T_A=\min\{(2^{59}\times T_1+T_2)\ 2^{g\times sec}, (2^{g\times sec}\times T_2+T_1)\ 2^{59}\}$$

Where T_A=impersonation time of attacker and *"g"* is the granularity level selected for CGA algorithm.

The following Lemma given by (Bos et al, 2009) shows that cost of second pre-image attack on a specific address:

$$T_A = \begin{cases} 2^{59} & if\ \sec = 0 \\ 2^{59+16\times sec} + 2^{16\times sec} & if\ 1 \le \sec \le 3 \\ 2^{59+16\times sec} + 2^{59} & otherwise \end{cases}$$

The address generation cost 'TG' is $T_G=2^{16\times sec}.T_2+T_1$and the address verification cost 'T_v' is $T_v=T_1+T_2$.

It is now obvious that the strength of CGA and its resistance against impersonation depends on the value of sec. Selecting a large sec value may cause unreasonable delay in address generation while as low sec value decreases security of algorithm.

5.5.3 CGA Limitations

The following subsection discusses about the drawbacks and limitation of the CGA algorithm.

i. Security Parameter or Sec Value

The security parameter "sec" is the predominant component in CGA algorithm impacting security and CGA generation time (Alsadeh et al,2012).With each increment of sec, the cost of CGA generation increases by a factor of 2^{16}.This increase has significant impact on mobile environment where handover operations need to be completed in few milliseconds to ensure adequate quality of service (Qadir & Siddiqi, 2011).Also the mobile devices are constrained by limited resources like battery, bandwidth, memory and processing power which need efficient utilization. In (Cheneau et al, 2010), authors found that sec value of only zero is feasible for mobile devices. The performance test was carried on Nokia N800 at the medium security level provided by 1024 bit RSA keys. Similarly in (Alsadeh et al, 2012), author's reported that on modern computing PC's; sec value greater than 1 is not feasible. To minimize the delay injected by sec value greater than 0, RFC 3972 makes two recommendations (Aura, 2005)

First, the steps [1-3] of CGA algorithm can be delegated offline or on separate more powerful computing machine. The result can later be ported to other nodes in the network. If a subnet changes, the nodes can use prior results for address generation. However, this approach turns

back our thinking to centralized model. Once this central server/machine is undermined, the whole network gets compromised.

Second, is to use small sec values depending upon the computational capacity of the node. The higher sec values can used in future when processing power of node increases.

ii. RSA Public Key Crypto System

The selection of public key crypto system plays an imperative role on the performance and governs computational cost and delay in CGA algorithm. Authors in (Cheneau et al,2010) report that increasing the RSA key size does not have large scale impact on CGA generation time, however RSA key generation time increases significantly when key size is increased.Thus, the total CGA generation time is substantially influenced by key size. The authors also in favor of substitute public key crypto system like ECC that will reduce the key generation time owing to smaller key size. Additionally it will also help in reducing packet size which is desirable in a low bandwidth scenario.

iii. Hash Function

The Hash function is another performance parameter impacting the level of CGA security and performance.CGA uses standard SHA-1(160 bit) hash function; however it is vulnerable to collision free attacks (Wang et al,2005).Thus, RFC 4982 introduced support for multiple hash algorithms in CGA (Bagnulo & Arkko,2007), however the RFC makes no recommendations of a hash function. (Lee & Mun, 2006) introduced the use of MD-5 hash instead of SHA-1 in their design of Modified CGA (MCGA).It was simpler and reduced execution time considerably. The use of MD-5 hash is not recommended due to its exposure to collision free attacks (Stevens, 2006).

iv. Global Time-Memory Trade-off Attack

Stated by lemma given in(Bos et al, 2009) a Global Time-Memory Trade-Off attack can be leveraged on CGA's in order to impersonate a node. To execute this attack, an attacker can generate a table of public/private key pairs or database of pre-computed interface identifiers from attackers own public keys. The attacker can than carry a profound search for hash collision. To mitigate this, CGA++ was proposed which includes subnet prefix in the calculation of

Hash2.The Modifier, Collision Count and Subnet Prefix are then signed by the private key of the owner. By this way, the attack cannot be applied globally because the attackers need to carry extensive search for each subnet separately. However the Time-Memory Trade-Off Attack is not an easy attack to execute and is practically infeasible. It's impractical to impersonate a node in a network due to large amount of storage required to carry the attack. (Bos et al, 2009) describes that in order to carry such an attack; the storage requirement is $2^{33-\min(n_i)}$ gigabytes, where *min* (n_i) denotes the smallest network size. In order to illustrate this, assume an attacker wants to impersonate a random node in network of size 2^{16}, this requires $2^{33-16} = 2^{17}$ gigabytes or 128 terabytes of storage. This is practically infeasible but not impossible.CGA++ also incurs additional cost because of digital signature verifications and requires additional time than CGA for the same sec value.

v. Privacy Implications

As a result of high computational nature of CGA creation, it's probable that once a node generates a valid CGA, it continues to use it in that subnet (Alsadeh et al, 2012).This exposes the node's interface to privacy based attacks. Using a fixed CGA for a longer period of time makes it possible for attacker to breach user's privacy by tracking the node. An attacker who sits in between the communicating node and the peer can correlate the data between the two. The correlation can be based on payload contents of the packets on the wire and the packet characteristics such as packet size and timing. An attacker can also access the communication logs of the peers with which the node has communicated. To avert this RFC 4941 recommends that addresses should be changed periodically (Narten, Richard & Suresh, 2007) to prevent attacker from collecting substantial amount of information about the network.

vi. DoS and Other Attacks

It is a well established fact that CGA's are vulnerable to DoS attacks (Nikander, James & Erik, 2004).An attacker can leverage the DoS attack on a node in one of the following ways (Alsadeh et al, 2012):

- DoS Attack against DAD-CGA
- DoS by Replay Attack
- DoS to kill a CGA node

In the first case, every time a node performs DAD on a tentative address, an attacker can reply with an acknowledgement indicating that address is already taken. After three tries, the victim node will automatically surrender and hence will be unable to configure the address. However; there are some potential limitations on the attacker's ability to leverage such an attack (Alsadeh et al, 2012). First, the attacker needs to have an access to the communication link and must be able to sniff the packets. Second, the attack needs to be leveraged quickly, i.e. attackers reply should be well before the end of DAD process. (Before Retrans timer value expires).As suggested in (Alsadeh et al, 2012), one possible solution to the above problem could be to discard the DAD with same tentative address, CGA parameters and signature. It's very unlikely that two CGA nodes in the network will be having same CGA data structure parameters. This is based on the birthday paradox given by (Bagnulo et al, 2002) which defines the probability of having at least two nodes in the network generate the same address:

$$Pb(n,k) \leq 1 - \left(\frac{n-k+1}{n}\right)^{k-1}$$

Where $n=2^{59}$ and k defines the number of interfaces on the link. For a large subnet of k=100,000 Pb (2^{59}, 100000) <=1.7e-08.This is a very small probability and thus validates the heuristics.

CGA's can also be abused by replay attack where an attacker can sniff the communication line and store signed messages from the victim. The attacker can replay the messages at a later point of time. The use of timestamp option with CGA is highly recommended to mitigate this attack.
A CGA node can also be flooded with valid or invalid messages sent with high frequency across the network. This keeps the destination nodes busy with verification process and thus wastes the computing resources.

5.5.4 Proposed Modifications to CGA

The following section describes the possible modifications that can be applied to standard CGA algorithm. These modifications significantly improve the performance of standard CGA and allow us to obtain a better transition model for enhanced CGA.

i. Imposing Time Constraint on CGA

Since the CGA's are strongly bound to the security parameter "sec" value, therefore having a higher sec value places no guarantee on the termination of brute force search. In order to avoid long CGA computational, it seems logical to place an upper bound on the address generation time. The idea was first conceived by(Aura & Roe, 2009) and later implemented by (Alsadeh et. al, 2012).The authors used time threshold as an input to CGA algorithm. If the threshold exceeds, the CGA algorithm terminates. The time parameter ensures that the brute force search stops after certain period of time. The most secure hash value can be chosen by selecting the hash value that has highest number of zeros in the leftmost bits of Hash2.To determine the security parameter "sec" automatically, the extension length can be rounded to the nearest multiple factor of "8" or "16" depending upon the granularity level selected. (Aura et al, 2009) argue that security parameters should be selected automatically instead of manual configuration because of the following two reasons. First, the impact on security and cost are on exponential scale, therefore it's difficult to set the parameters correctly. Selecting low sec values can result in weak hashes where as high sec values can cause the brute force search for suitable modifier to take hours or years. Second, it's vexatious for a user to comprehend the complexities of CGA and to update the parameters regularly. Thus it seems logical to set the extension length automatically to an optimal value. The time threshold also depends on external factors like user allotted time frame for address generation or how many hashes a CPU is expected to compute in a given time frame.

Although it's fairly easy to implement the time threshold condition in CGA, it also induces performance bottlenecks and leads to wastage of CPU resources (Aura et al, 2009). For example ,if time threshold=60 seconds is chosen and if a modifier with sec=1 is found after one second of the brute force search, its unproductive to continue the search for the remaining 59 seconds. Therefore it seems logical to end the brute force search early when the probability of finding a modifier for a next higher security parameter value in the allocated time falls below a specific

threshold (Aura et al,2009).The algorithm demands a general rule for stopping the brute force search.

ii. Reducing Granularity Factor

The time constraint on CGA may waste the CPU resources because the granularity factor "16" is quite large (Alsadeh et. al, 2012; Aura et al,2009).The multiple factor "16" was chosen so as to increase the maximum the length of hash extension up to 112 bits. This allows the extended hash length to grow up to 59+ (16×7) =171 bits. The benefits of this are questionable keeping in mind the current CPU capabilities and many parts of the security infrastructure probably depend on 128 bit one way hashes. Normally, hashes of length greater than 80 bits are considered secure.(Aura et al,2009) suggest that using multiple factor of "8" could be a more sensible and feasible option. The probability of having "8" successive zeroes is more than "16" successive zeroes in brute force search. This increases the maximum length of hash extension up to 56 bits. Thus, total length of the hash varies between 59 and 115 bits i.e. (59+ (8×7)) which is feasible for current CPU speeds. Smaller granularity factor is also suitable for mobile devices which have limited computational power.

iii. Including Subnet Prefix

As discussed above, Global time-memory trade-off attacks can be executed on CGA's. To diminish this attack, (Bos et al,2009) suggests that subnet prefix should be included in the calculation of Hash2.This has two advantages. First, the security is increased against the time-memory trade-off attack because the attacker needs to generate a separate database for each subnet. Second the node privacy is maintained because it has to generate a new CGA for each subnet. However the inclusion of subnet prefix introduces some efficiency loss. In a mobile environment, a node frequently changes its subnet. To search for a valid modifier every time is an infeasible option in mobile devices as they have constrained computational power. It was design consideration of (Aura, 2005) to not include subnet prefix for the sake of efficiency. However the computational of computing devices is going to increase in near future. The commonly cited example is the Moore's law which states that the speed of computation doubles every 12-24 months.

iv.Replacing RSA with ECC

The current standard CGA uses RSA cryptosystem to generate public key and signature. This signature and other CGA parameters enable the receiver to validate the relationship between the CGA address and public key. Minimizing the size of CGA parameters will greatly help reduce the bandwidth consumption.(Cheneau et al,2010) argues that using an alternate cryptosystem like Elliptic Curve Cryptography (ECC) or Elliptic Curve Digital Signature Algorithm (ECDSA) is efficient for resource constrained networks like mobiles, sensor nodes etc.Using ECC/ECDSA instead of RSA in CGA offers the following advantages:

- ECC/ECDSA takes less time for computation of the signature.
- ECC key size is shorter as compared to RSA key size for the same security level. Shorter key size helps reduce bandwidth consumption and thus is suitable for small capacity devices.

Table 5.1 shows the equivalence between ECC and RSA key lengths in terms of security level

RSA Key Length (bits)	**ECC key length (bits)**
1024	160
2048	224
3072	256
7680	384
15360	512

Table 5.1 RSA and ECC Equivalence (NIST, 2007)

v. Replacing Hash Function

Another important parameter that affects the CGA security is the hash function. Standard CGA uses SHA-1, however due to security flaws as described in (Wang et al, 2005; Bagnulo et al, 2007), it will soon be phased out. The need of the hour is the transition to more secure hash function like SHA-256 orSHA-512.Although, these hash functions tend to be slower than SHA-1, but offer robust and flawless security.

5.5.5 Implementation and Evaluation

The modified CGA is run on a machine having windows 7 operating system installed. The machine software is powered by Intel core i5 processor clocked at 2.53 GHz speed. The machine has installed 4 GB of DDR2 of RAM. We coded the standard CGA as well as modified CGA algorithm in C#.NET due to flexibility of language and rich inbuilt function support. This has some efficiency loss, but still provides an easy and fair platform for comparison.

The algorithm in addition to CGA parameter data structure takes subnet prefix as input in the calculation Hash2.This ensures security against global-memory trade-off attacks and limits the scope of attack against the CGA node. We also use granularity factor of "8" instead of "16" because probability of having "8" successive zeroes is more than "16" successive zeroes in brute force search. ECC cryptosystem is used for public key generation because of its computational benefits as discussed above. For hash calculation, SHA-1 which is believed to have security vulnerabilities is replaced by SHA-256. In modified CGA, we do not use the time threshold condition as it leads to wastage of CPU resources (Aura et al, 2009).Both the standard CGA and modified CGA algorithms are executed 50 times and average value is calculated.

Table 5.2 shows the time measurement for CGA and modified CGA generation algorithm. This includes the total time from the generation of public key up to the computation of interface identifier(including Hash1 and Hash2).The table also shows the CGA and modified CGA verification time, however it does not include RSA digital signature verification. Standard CGA uses granularity factor of 16 while as modified CGA uses granularity factor of 8.Table 5.2 also shows comparison based on the cryptosystem used. While the standard CGA uses RSA for public key generation, the Modified Version uses ECC public key cryptosystem.

Sec Value=0				
RSA Key Length	**1024**	**2048**	**3072**	**7680**
CGA Generation time	0.456	1.91	7.11	233.4
CGA Verification Time (milliseconds)	2.56	2.57	2.6	2.6
Equivalent ECC Length	**160**	**224**	**256**	**384**
Modified CGA Generation Time	0.012	0.15	0.87	3.2
Modified CGA Verification Time (milliseconds)	1.9	2.0	2.0	2.0
Sec Value=1				
RSA Key Length	**1024**	**2048**	**3072**	**7680**
CGAGeneration time	1.2	2.22	7.98	344
CGA Verification Time(milliseconds)	2.56	2.67	2.87	2.88
Equivalent ECC Length	**160**	**224**	**256**	**384**
Modified CGA Generation Time	0.67	0.99	1.2	1.33
Modified CGA Verification Time (milliseconds)	1.8	2.33	2.33	2.34
Sec Value=2				
RSA Key Length	**1024**	**2048**	**3072**	**7680**
CGA Generation time	-----	-----	-----	-----
CGA Verification Time (milliseconds)	-----	-----	-----	-----
Equivalent ECC key Length	**160**	**224**	**256**	**384**
Modified CGA Generation Time	1.23	2.23	7.77	512
Modified CGA Verification Time (milliseconds)	3.32	3.33	3.45	3.45

Contd…

Sec Value=3				
RSA Key Length	**1024**	**2048**	**3072**	**7680**
CGA Generation time	-----	-----	-----	-----
CGA Verification Time	-----	------	-----	-----
Equivalent ECC key Length	**160**	**224**	**256**	**384**
Modified CGA Generation Time	2.56	2.56	3.12	5.15
Modified CGA Verification Time (milliseconds)	3.44	3.44	3.45	3.67

Table 5.2 CGA/Modified CGA Time Comparison (in seconds)

Analyzing table 5.2 we see that Modified CGA significantly improves computational time as compared to standard CGA; however there is no significant impact on CGA/Modified CGA verification time. As seen in table 5.2,Modified CGA generation time using 256 bit ECC key with sec=1 is shorter than generation time using a 3072 bit RSA key for sec=0.This infers that ECC increases the security level while reducing the generation time. Also Modified CGA allows us to use the value of security level sec=3. i.e. 24 zeroes; while as standard CGA computation beyond sec=1 becomes infeasible.

One of the major benefits of using ECC cryptosystem is that it takes less time for computation as compared to RSA.Also ECC key size is shorter as compared to RSA key size for the same security level which results in less bandwidth consumption. In CGA, the length of the data structure has a direct impact on the computation time of hash digests. Table 5.3 extracted from (Cheneau et al, 2010)reports the size of CGA parameter data structure in bits using RSA and ECC and the number of generated 512 bit blocks for different key lengths. The hash computation time is constant for every block and complexity of the computation is bound to the number of blocks. Also low size of CGA data structure will help in reducing packet size which is desirable in a low bandwidth scenario. From table 5.3, we see that Hash1 and Hash2 generation time of CGA is greater with RSA keys than ECC keys. This is because RSA generates more blocks as compared to ECC.Using ECC results in lower size of CGA data structure which help conserve bandwidth.

Using RSA Cryptosystem				
RSA Key Length	**1024**	**2048**	**3072**	**7680**
DER Encoded RSA public key length (bytes)	160	292	420	996
CGA parameter data structure length (bits)	1480	2536	3560	8168
Number of 512 bit blocks	4	6	8	17
Using ECC Cryptosystem				
ECC Key Length	**160**	**224**	**256**	**384**
Octet Encoded Public Key Length (bytes)	66	80	88	120
CGA parameter data structure length (bits)	728	840	904	1160
Number of 512 bit blocks	2	2	2	3
Number of bits saved in ECC (bits)	752	1696	2656	7008

Table 5.3 Length of CGA parameter Data Structure

5.6 SEND Deployment Challenges

Though SEND addresses most of the security vulnerability issues with Neighbor Discovery, however it does not embrace the confidentiality factor and the link layer security (Arkko et al, 2005).SEND has a number of limitations and drawbacks which impede its deployment and integration into current IP networks. First, the CGA option can't guarantee about the real node's identity and it's not sufficient enough to prove that the CGA address is used by a legitimate node (Alsadeh et al, 2012).As CGA's are not certified, any malicious node could create a valid CGA from its own set of public keys and start the communication. However; only the legitimate user can claim ownership and sign NDP message by its private keys. Thus an attacker may

impersonate other node's CGA but cannot sign the messages from that node. Second, CGA's are themselves vulnerable to DoS attacks against CGA-DAD messages. Attackers can successfully use non-CGA and respond to every CGA-DAD message with claiming an address. As the value of collision count exceeds 2, CGA generation algorithm fails. Attackers can also fail the CGA by capturing ND messages and altering the senders CGA parameters so that the CGA verification process at the destination may fail. In addition to the above, standard CGA is vulnerable to Global Memory Trade Off attack on nodes and attacks on router authorization. SEND protocol also suffers from privacy constraints. Due to high computational cost of CGA, it's likely that a node which generates an acceptable CGA continues to use that in the network. Therefore these nodes can be tracked down in course of time and are thus vulnerable to privacy attacks. However this issue can be resolved by assigning a lifetime to an address (Alsadeh et al, 2012). After the timer expires, CGA needs to be discarded so that new CGA could be generated. Most of the operating systems today support Neighbor Discovery protocol; however lack mature implementation of SEND protocol in the system. Current SEND implementations for specific OS distribution, such as Debian Linux, are basically proofs of concept rather than production-ready software(Alsadeh et al, 2012).This unavailability of protocol impedes deployment and acceptance of IPv6 in current systems.

5.7 Summary

CGA's are promising technology for providing authentication in a decentralized way. CGA's are computational dependent on security parameter (sec) which impact the CGA generation time. The standard CGA uses RSA public key cryptosystem with SHA-1; however its use is not feasible for constrained devices. Also the granularity factor of "16" is questionable with current CPU speeds. In this chapter, we evaluate the performance and discuss techniques that can be used in optimizing the use of IPv6 CGA. The techniques discussed were the modification approaches to the standard RFC 3972 for CGA optimization. First step towards optimization is to reduce the granularity factor from "16" to "8" which ensures that brute force search terminates quickly.

Also, ECC cryptosystem must be used instead of RSA which is efficient for resource constrained devices. This minimizes the size of CGA data structure which help conserve bandwidth. The modified algorithm also uses SHA-256 hash function instead of SHA-1 which is known to have

security vulnerabilities. The modified CGA provides protection against global-time memory-tradeoff attacks by including subnet prefix in the calculation of Hash2.

The chapter evaluated the modified CGA and compared it with standard CGA. The results show that modified CGA significantly improves computational time and is more secure as compared to standard CGA. Although these approaches show improvement in generation time of CGA, they are still computation intensive and need significantly to be improvised. Also CGA are vulnerable to privacy attacks and lack authentication. Future work needs to be done to make CGA more secure and deployable for resource constrained devices like mobile phones.