Improving Cryptographically Generated Address Algorithm in IPv6 Secure Neighbor Discovery Protocol through Trust Management

M. Moslehpour, S. Khorsandi

Open Science Index, Humanities and Social Sciences Vol:10, No:6, 2016 publications.waset.org/10004519.pdf

Abstract**—**As transition to widespread use of IPv6 addresses has gained momentum, it has been shown to be vulnerable to certain security attacks such as those targeting Neighbor Discovery Protocol (NDP) which provides the address resolution functionality in IPv6. To protect this protocol, Secure Neighbor Discovery (SEND) is introduced. This protocol uses Cryptographically Generated Address (CGA) and asymmetric cryptography as a defense against threats on integrity and identity of NDP. Although SEND protects NDP against attacks, it is computationally intensive due to Hash2 condition in CGA. To improve the CGA computation speed, we parallelized CGA generation process and used the available resources in a trusted network. Furthermore, we focused on the influence of the existence of malicious nodes on the overall load of un-malicious ones in the network. According to the evaluation results, malicious nodes have adverse impacts on the average CGA generation time and on the average number of tries. We utilized a Trust Management that is capable of detecting and isolating the malicious node to remove possible incentives for malicious behavior. We have demonstrated the effectiveness of the Trust Management System in detecting the malicious nodes and hence improving the overall system performance.

Keywords**—**NDP, SEND, CGA, modifier, malicious node.

I.Introduction

To respond the shortage of IPv4 address space, transition from IPv4 to IPv6 has been happened. On 8th June 2011, websites and Internet Service Providers (ISP) around the world joined together in World IPv6 Day for a successful global-scale trial of the new Internet Protocol, IPv6. On 6th June 2012, major ISPs, home networking equipment manufacturers, and web companies permanently enabled IPv6 for their products and services.

Although migration to IPv6 was indispensable, the business sector needs to do it in a secure manner in order to avoid the possible security risks inherent in an IPv6 deployment.

The NDP are used by IPv6 nodes for several functions like discovery of routers and nodes on the link, finding the mapping between the Media Access Control (MAC) address and the link local addresses, detecting duplicate addresses, and maintaining reachability information about the paths to active neighbors. Although the NDP has some critical functions, it is vulnerable to certain attacks such as spoofing, Denial of Service (DOS), Replay, Redirect and Rogue router attacks because it does not include any security provisions and it was designed to work in trustworthy links where all nodes on the link trust each other. But in reality, all nodes on a link cannot be trusted. For instance, in public networks a malicious user can forge NDP messages and generate attacks by impersonating legitimate nodes [1]. These attacks are Neighbor Solicitation (NS)/ Neighbor Advertisement (NA) spoofing, Neighbor Unreachability Detection (NUD) Failure, Duplicate Address Detection (DAD), DoS, Malicious Last Hop Router, Spoofed Redirect Message, Bogus On-Link Prefix, Parameter Spoofing, and Replay attack [2]. To mitigate these attacks and enhancing the security of IPv6 neighbor and router discovery, SEND was proposed [3]. By SEND, message integrity is ensured, the authority of routers is verified, and IPv6 address theft and Replay attacks are prevented. Four new options can be appended to the regular NDP message in order to create a SEND packet: CGA, RSA signature, Nonce, and Timestamp [1]. A public-private key pair must be generated or obtained by a SEND-enabled node before claiming an address. Then the SEND-enabled node generates the CGA address based on the public key and other auxiliary parameters. The private key is used to sign the outgoing ND messages from that address. The SEND verifier node checks the received address by calculating a hash of the corresponding public key and that the signature. If both verification steps are successful, then the received address is known as a valid address. The Router authorization is done by a certificate that it gets from a trust anchor.

CGAs are IPv6 addresses such that the Interface Identifiers are generated by one-way hashing of the node’s public key and other auxiliary parameters. This consist fundamental parts of the SEND. Although SEND protects NDP messages from some attacks, it has several pitfalls such as its computation. In standard CGA generation, two independent one-way hash values (Hash1 & Hash2) are computed. The Hash2 calculation determines an input parameter for the Hash1 calculation. The purpose of the Hash2 (the second hash) is to increase the computational cost to the hacker for doing a brute-force attack, without increasing the length of hash output value. Applying CGA algorithm will not be reasonable, if “Sec” value is not zero because of high computational cost of Hash2 value.

We apply the idea of computing CGA in multi-node base. In this idea, we use resources of the computer network by dividing the CGA computation on some nodes of the network.

Mahnaz Moslehpour is with the Amirkabir University of Technology (Tehran Polytechnic), Iran (e-mail: moslehpour@aut.ac.ir).

Siavash Khorsandi is with the Computer Engineering and Information Technology Department of Amirkabir University of Technology (Tehran Polytechnic), Iran (e-mail: khorsandi@aut.ac.ir).

The benefit of this new approach is that if nodes of network have single core CPU, the time of computing CGA will decreases and it is less than the time computed with [4] because the final modifier value is calculated faster. Also, if nodes have multi-core CPU, this time will decrease to a smaller value significantly. Also, if one node stops suddenly, its task to compute the final modifier will transfer to other nodes. Furthermore, we focus on the influence of the existence of malicious nodes on the overall load of un-malicious ones in the network and on the time of CGA generation.

This report is structured as follows: Overview of NDP and Secure NDP are represented in Chapter II and chapter III, respectively. Chapter IV described CGA. In Chapter V, we studied some previous research that was done. Chapter VI describes our proposed solution and its implementation. We expressed the significance of the method in Chapter VII. Finally, in Chapter VIII, the conclusion and future works are mentioned.

II. Neighbor Discovery Protocol

According to RFC 4443, Neighbor Discovery (ND) is one of the most important functions of the Internet Control Message protocol for IPv6 (ICMPv6). Its messages are implemented as a set of ICMPv6 Types and options and follow the ICMPv6 message formats. NDP massages consist of an ICMPv6 header, ND message specific data, and zero or more options [5]. This protocol performs functions that are similar to those addressed by the ARP and ICMP, as well as Router Discovery and Router Redirect protocols used in IPv4. NDP defines five ICMPv6 packet types:

1. Router Solicitation-Type 133: The Router Solicitation message is sent by IPv6 hosts to discover the presence of IPv6 routers on the link.
2. Router Advertisement-Type 134: IPv6 routers send unsolicited Router Advertisement messages periodically and solicited Router Advertisement messages in response to the receipt of a Router Solicitation message.
3. NS- Type 135: It is used by nodes to determine the link layer address of an on-link IPv6 node, or to confirm the reachability of the node.
4. NA - Type 136: NAs are sent by nodes to respond to an NS message.
5. Redirect- Type 136: The Redirect message is sent by an IPv6 router to inform an originating host of a better first hop address for a specific destination.

These messages are used to provide the following functionality:

1. IPv6 address to MAC address resolution: Like ARP in IPv4, One of the duties of NDP in IPv6 is resolution of IPv6 address to MAC address.
2. Router discovery: Discovering routers in an IPv6 network using Router Solicitation & Router Advertisement messages.
3. Prefix discovery: Discovering IPv6 network prefixes where the host belongs to, by using Router Solicitations & Router Advertisement messages.
4. Maximum Transmission Unit (MTU), hop limits: Parameters such as the hop count and MTU are listed in the Router Advertisement (RA) message sent by the router.
5. DAD: NDP is used to detect whether duplicate IPv6 addresses exist in an IPv6 network.
6. NUD: Detecting that a neighbor is no longer reachable.
7. Next-hop determination: to send a packet, Next-hop determination is the first task that any host performs. This can be a router or the destination itself. This algorithm is used to map an IP destination address into the neighbor's IP address to which traffic for the destination should be sent.

As both hosts and routers use NDP, it is vulnerable to various attacks. These attacks are Attacks on ND, DAD DoS Attack, NUD Failure, Parameter Spoofing Attack, Bogus Address Configuration Prefix, Bogus On-Link Prefix, Malicious Last Hop Router Attack, Kill the Default Router Attack, Compromise of a Router, Spoofing of Redirect Messages, and Remote/Replay Attacks [6].

III. Secure ND

The SEND protocol is designed to counter the threats to NDP. SEND is applicable in environments where physical security on the link is not assured (such as over wireless) and attacks on NDP are a concern. SEND offers address ownership proof, message protection, and a router authorization mechanism features as additional ones. In order to achieve these enhancements, SEND encodes its messages by creating new Option Types in ICMPv6. To create a SEND packet, four new options can be appended to the regular NDP message; CGA, RSA Signature, Nonce, and Timestamp [1].

IV. CGA

One of the basic foundations of SEND is CGA that was introduced, in order to prevent against IP address spoofing and stealing attacks. CGA are IPv6 addresses for which the interface identifier is generated by computing a cryptographic one-way hash function using public key and auxiliary parameters of a host [1]. This ensures that the IPv6 address of the host is bound to its public key. CGA has two main processes: 1) Generation, 2) Verification. The Generation process is started after determining the address owner's public key and choosing the appropriate Sec value, this process is starting and by finding the Final Modifier it is finished. After that the Hash1 computation begins. The output of Hash1 computation is Interface Identifier (IID). By concatenating the IID and Subnet prefix, as the final step, the DAD process is done for ensuring that there is no address collision within the same subnet. If address conflict does occur, then the collision Count will be incremented and the Hash1 process will be repeated until a link-unique address is obtained. The Verification process is done when the hash value is re-computed and compared with the interface identifier of the sender's address [7] in order to determine whether an attacker impersonates an existing IPv6 address [8].

V.Literature Review

One disadvantage of CGA algorithm, as stated earlier, is its high computational cost. This issue encourages researchers to look for less time-consuming protocols yet optimized solutions. Some of such studies are:

1) Replacing ECC keys with RSA keys to improve performances of the CGAs; because the lengths of the ECC key are small then the CGA generation time decreases. By replacing ECC key, the CGA generation time is decreased in few milliseconds; this milliseconds decrease is significant in networks (e.g. mobile) where resources are limited [9].
2) Configuring CGAs using DHCPv6; assigning some parameters to hosts and managing the use of CGAs is done by the network management. In a DHCPv6 managed network, the DHCPv6 server is needed when a host may begin a request for the relevant CGA configuration information and the server responds the host by sending the configuration information [10].
3) Optimistic DAD for IPv6. IPv6 address configuration mechanisms provide appropriate collision detection mechanisms for the fixed hosts. While, by increasing a number of nodes in a network, these nodes need to maintain continuous network access despite changing their network attachment. To fast address configuration, Optimizations to the DAD process are necessary [11].
4) WinSEND: Windows SEcure Neighbor Discovery. Hash2 is computed to find an appropriate final modifier. This computation process is the most expensive part of CGA generation algorithm. In order to speed up this process, the generation algorithm is parallelized. WinSEND is used to do the brute-force search to satisfy Hash2 condition of CGA algorithm. In parallel mode, WinSEND can use the whole CPU capacity to finish CGA computations [12].

VI.Proposed Method

To solve the limitation of previous researches, the idea is utilizing the network resources to compute the Hash2 value. When a node finds the appropriate modifier value (where the 16*sec bit value of it is equal to zero), it sends a broadcast message to other nodes and notifies them to stop their computation.

A. *Trust Management*

To implement the above idea, we use a tool that has been developed in Microsoft.Net as a service to provide security for Windows NDP. This tool is applicable in all Windows Family [13]. It uses Winsock library to transfer data between Network Interface Card (NIC) to the upper layers and vice versa [4]. Winsock API is implemented in windows operating systems in order to access network services specially TCP/IP.

In un-Trusted network that the probabilities of existence of malicious nodes are high, a system that can detect malicious nodes and manage them is needed. Trust Management System isolates malicious nodes due to remove possible incentives of malicious behavior.

1) Trust Management Algorithm

When a value that is calculated by each node receives to new node, it verifies this value based on the conditions of 16*sec bit value = 0 and a public key of the sender.

If each of these two steps fails, the calculated value is discarded. The result of previous steps helps to determine the state of participated nodes. Based on the state of nodes, jobs are assigned to them till appropriate value is calculated.

Some of the functionality of this system is follows:

1. Determining the state of available nodes that participate in the computation process based on the responses that each node sends.
2. Informing other nodes about the state of other node(s).
3. Isolating (a) node(s) with Black state. Other nodes discard messages that are sent by Black-state node(s).
4. Assigning no job to the Black-state node(s). In this case, other nodes have to perform jobs.
5. After 5 seconds isolation, a job is assigned to (a) Black-state node(s).

The first task of Trust Management System is determining the state of participated nodes. There are three different states according to the operation of nodes: 1) White State: a node with this state, always return appropriate computed value and the sender information can trust it. This node always participates on computing the Hash2 value. 2) Gray State: a node with this state sometimes computes the appropriate value and sometime does not. In some cases, a node with Gray state, may act as a malicious node or an honest node. If it returns an (in) appropriate value after getting some jobs, its state will change. At beginning of the computation, all participating nodes in computing the Hash23 value are in this state. 3) Black State: a node with this state, always compute inappropriate value. After a node that its state is Gray returns N wrong computed values continuously, in this experiment we set N=5, its state are changed to Black and the original node assumes this node is malicious and isolates it. In this case, the original node sends a message to other nodes and notifies them that this node with a specific IP address is malicious node. So, other nodes do not communicate to this node anymore and they discard requests from this node.

After specifying the state of participate nodes in computation process, task of computing appropriate modifier is distributed between nodes.

In a Trusted network, the load of each node is almost equal because all of them are in the White-State and don’t behave maliciously. Therefore, the load of each node is almost equal. However, in Un-Trusted network, by increasing the number of malicious node(s), load of honest nodes is increased; as long as nodes do not compute the appropriate modifier value, the computation process is continued. In this case, when the state of nodes is changed, the load of other nodes is changed too; for instance, if 1 out of 3 nodes is malicious, before detecting the state of that node, it takes task of computing, but when its state is modified to black state, its task is assigned to other nodes and they have to continue computation. This process is continued till the end of computation process.

Open Science Index, Humanities and Social Sciences Vol:10, No:6, 2016 publications.waset.org/10004519.pdf

2) Success Ratio of Detecting Black-State Nodes

Besides all the benefits of Trust management system, one of its disadvantages is detecting false nodes' status. In this case, a malicious node that computes an inappropriate modifier value is not isolated by Trust Management system and will remain in its previous state or is changed to false state. So, it continues its malicious behavior. Furthermore, the state of a white state node might be changed to black-state which causes the generation time and overall load of other nodes are increased.

To calculate success rate in detecting state of nodes, we need to define criteria. In this study, Dis-Honesty Percentage of nodes specifies the total percentage of the dishonest nodes; and dishonesty percentage of each node is not altered by changing its state. With this assumption, the ratio of black-state nodes to all nodes is the success rate of detection of nodes' state:

$$\frac{Total\ number\ of\ black\ state\ nodes}{Total\ number\ of\ Un-Honest\ nodes} = \frac{7}{12} * 100 = 58.3\ \% \qquad (1)$$

According to (1), the success rate of the system in detection of malicious nodes is 58.3%.

VII. Presentation, Analysis and Interpretation of Data

A. Implementation Scenarios

This study is implemented in an Un-Trusted network that there are honest and dishonest nodes. A new node cannot sure the received responses are correct because dishonest nodes act maliciously. In this kind of network, malicious nodes can send wrong responses and disrupt the other nodes; And Node-Based structure that distribution information to nodes and evaluation of the nodes' responses are done by a new node who wants to connect to the network.

The tool is extended to do the brute-force search to satisfy Hash2 condition of CGA algorithm in parallel. In this mode, the extended tool can use almost whole CPU capacity to finish CGA computations. Based on the number of CPU cores of nodes, the numbers of parallel tasks which can be used for CGA computations are determined.

The evaluation criteria are, 1) Average Tries, 2) Average Generation Time, 3) Overall Load and 4) Dis-Honesty percentage.

To evaluate the performance of CGA generation algorithm in parallel mode, several experiments are carried out for different systems specification. The experiments are done on a base computer with 2.00 GHz CPU (4 cores). Windows 7 is the main operating system on this computer. We run the extended tool on guest windows 7 (32-bit) hosted by VMware Workstation 9.0 software. The settings of VMware Workstation offer the flexibility to control the number of virtual CPU cores that the guest operating systems can use. The CGA generation process is generated 1000 times to have sufficient samples because this process is a random process, and there are no guaranties when it will stop. All the measurements for different number of cores are taken for CGA with Sec value "1" and with key size 1024-bit.

B. Test Evaluation

We test the extended tool in different situation. The tables below show the results of these experiments are done with and without trust management system. Tables I and II show the CGA Average Tries, Average CGA generation time, and Overall load with 3-cores.

TABLE I
TEST RESULT IN A NETWORK WITH TRUST MANAGEMENT SYSTEM

Dis-Honest Nodes (Percentage)	Average Tries	Average Gen. Time	Overall load
10%	33531.224	239.183	0.229
20%	44070.166	262.452	0.530
30%	54081.842	289.623	0.803
40%	63081.065	301.468	0.892
50%	64616.480	305.897	0.934
60%	67531.343	315.167	0.954

TABLE II
TEST RESULT IN A NETWORK WITHOUT TRUST MANAGEMENT SYSTEM

Dis-Honest Nodes (Percentage)	Average Tries	Average Gen. Time	Overall load
10%	57343.748	273.117	0.812
20%	64941.046	280.392	0.970
30%	69163.905	320.445	0.989
40%	69892.181	340.065	0.991
50%	69928.967	343.996	0.992
60%	70011.158	352.893	1.000

Tables I and II show that in the case of applying Trust management system, when Trust management system is used, the worst value of overall load parameter is approximately 0.95. However, the value of this parameter when the Trust management system is not used, is 1; we can conclude that in this case, the overall load of nodes in network is in the worst state. Also, the value of this parameter when 30% of nodes are malicious and the Trust management system is applied is almost worth the value of it in which 10% of nodes are malicious in lack of Trust management system in the network. Also, the value of Average Generation time when 60% of nodes are dishonest while trust management system is applied approximately equals to the value of Average Generation time when 30%of nodes are malicious and there is no trust management system. This issue is true in the case of Average Tries parameter. In comparison of having and not having the Trust Management System, the Average Tries parameter when 30% of nodes are malicious is lower than when 10% of nodes are malicious.

In general, according to Tables I and II, the percentage of malicious nodes, Average Tries, Average Generation time, and overall load parameters have a direct impact on each other. In other words, by increasing the number of malicious node (since these nodes calculate inappropriate values) the overall load of network and the tries of un-malicious nodes will be increased and as a result, the time of generation CGA is increased as well.

It is concluded that if the number of malicious nodes decreases and it approaches zero, the value of Generation time and tries would decreases.

Fig. 1 demonstrates the Average Generation Time parameter in two states of applying Trust management system and without using it. Fig. 2 demonstrates the Overall Load parameter in two states of applying Trust management system and without using it.

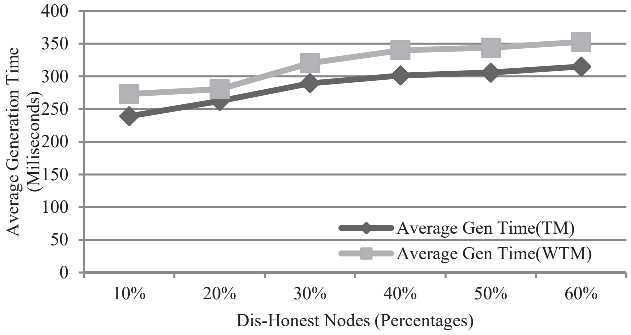

Fig. 1 The graph of Average Gen. Time parameter on Dis-honest nodes in both cases of using Trust Manager and without applying it

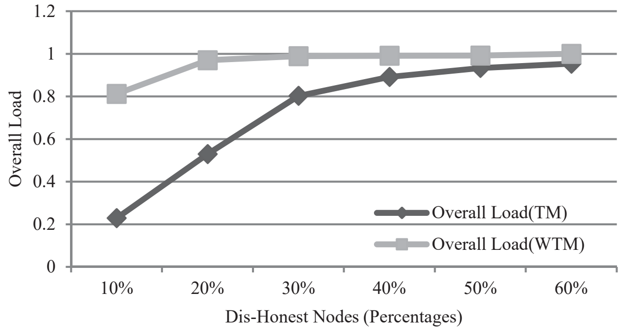

Fig. 2 The graph of overall load parameter on Dis-honest nodes in both cases of using Trust Manager and without applying it

By increasing the percentage of malicious nodes, the amount of Average Generation Time and Average Tries will be increased. Also, the number of malicious nodes has the direct impact on overall load. By increasing or decreasing this value, the amount of overall load parameter will be increased or decreased respectively.

VIII.Conclusion and Future Work

The goal of this research is to design and implement a system that can minimize the computation time of the CGA generation process and makes it cost-effective for all users who want to connect to the Send-Enabled network.

The CGA generation process has a wide range that each part of this process can be considered and restated with novel ideas. Some ideas are discussed as following:

In SEND, an address generator does brute force search on different values for the Modifier until the condition of 16*Sec-leftmost bits of Hash2 computes to zero. Large Sec value leads to significant and undesirable address generation delay and in networks, such as in cellular networks, and wireless sensors where nodes have limited resources, it is impractical. Therefore, the high computation cost of CGA may prevent its usage and leave IPv6 network vulnerable to some attacks which are related to address stealing. To decrease the high computation cost of CGA, finding a solution that can be used in the recourse-constrained networks instead of applying the condition of 16*Sec-leftmost bits of Hash2 equals to zero that provides the same security level, is efficient.

In addition to the problem of computational time of the CGA process, it is also vulnerable to some attacks and so far there is no solution for that. Studies on this issue and finding a solution to protect CGA from these attacks make it more efficient.

References

[1] A. AlSa'deh, H. Rafiee, and C. Meinel, "Secure Neighbor Discovery: A Cryptographic Solution for Securing IPv6 Local Link Operation", Theory and Practice of Cryptography Solutions for Secure Information Systems Book, May, 2013.

[2] P. Nikander, J. Kempf, and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756 (Informational), Internet Engineering Task Force, May 2004.

[3] J. Arkko, J. Kempf, B. Zill, and P. Nikander, "Secure Neighbor Discovery (SEND)", RFC 3971, Internet Engineering Task Force, March 2005.

[4] H. Rafiee, A. AlSa'deh, and Ch. Meinel, "Multicore-Based Auto-Scaling Secure Neighbor Discovery for Windows Operating Systems", 26th IEEE International Conference on Information Networking (ICOIN), February 2012.

[5] A. Conta, S. Deering, M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, Internet Engineering Task Force, March 2006.

[6] A. Raghavan, "Secure Neighbor Discovery: A Report".

[7] S. Qadir and M.U. Siddiqi, "Cryptographically Generated Addresses (CGAs): A Survey and an Analysis of Performance for Use in Mobile Environment", IJCSNS International Journals of Computer Science and Network Security, February 2011.

[8] T. Aura, "Cryptographically Generated Addresses (CGA)". RFC 3972, Internet Engineering Task Force, March 2005. Online Available: http://tools.ietf.org/pdf/rfc3972.pdf.

[9] T. Cheneau, A. Boudguiga, M. Laurent, "Significantly Improved Performance of the Cryptographically Generated Addresses Thanks to ECC and GPGPU", Computers and Security, Elsevier, 2010.

[10] S. Jiang & Z. Xia, "Configuring cryptographically generated addresses (CGA) using DHCPv6".Retrieved from http://tools.ietf.org/html/draft-ietf-dhc-cga-config-dhcpv6-02, 2012.

[11] N. Moore, "Optimistic duplicate address detection (DAD) for IPv6". RFC 4429, Internet Engineering Task Force, September 2006.

[12] H. Rafiee, A. AlSa'deh, and Ch. Meinel, "WinSEND: Windows SEcure Neighbor Discovery", 4th International Conference on Security of Information and Networks (SIN 2011), 14-19 November 2011, Sydney, Australia 2011.

[13] OS Platform Statistics, http://www.w3schools.com/browsers/browsers_os.asp, 2011.