Using SEND signature algorithm agility and multiple-key CGA to secure proxy neighbor discovery and anycast addressing

Tony Cheneau, Maryline Laurent

Using SEND Signature Algorithm Agility and Multiple-Key CGA to Secure Proxy Neighbor Discovery and Anycast Addressing

Tony Cheneau, Maryline Laurent
Institut TELECOM, TELECOM SudParis
CNRS Samovar UMR 5157
9 rue Charles Fourier, 91011 Evry, France
firstname.lastname@it-sudparis.eu

***Abstract*— The Neighbor Discovery Protocol (NDP) is a fundamental component of the IPv6 protocol suite in charge of the Link-layer interactions (Address Resolution, Router Discovery, etc.). Over the years, it has been extended to new usages, such as Mobility (Mobile IPv6), proxy advertisements (Neighbor Discovery Proxies) and security (Secure Neighbor Discovery, SEND). However, SEND's protection is currently incompatible with two NDP functions, namely the proxy Neighbor Discovery function (used in Mobile IPv6) and the IPv6 anycast addresses (i.e. shared addresses on a same link).**

On one hand, Cryptographically Generated Addresses (CGA) and SEND protect the NDP messages. The former, an address generation scheme, binds a single public key to an address. The latter secures NDP messages by signing them with the corresponding private key of the source address, thus achieving a proof of address ownership. On the other hand, proxy Neighbor Discovery and IPv6 anycast addressing are mechanisms binding one address to multiple nodes.

In this article, we present an overview of the existing solutions addressing these divergent objectives and tackle their limitations. We then propose an alternate solution and introduce the Multiple-Key Cryptographically Generated Addresses (MCGA) concept. This proposal relies on SEND's Signature Algorithm Agility extensions (also defined by the authors) to bind more than one Public Key to an address. As such, it enables multiple nodes to properly share and protect the same address and thus resolves proxy Neighbor Discovery and Anycast issues. Finally, we present implementation results and discuss the advantages of our approach over the existing solutions.

I. Introduction

In the IPv6 environment, the Neighbor Discovery Protocol [1] (NDP) is the successor of the venerable Address Resolution Protocol (ARP). While it retains most of its predecessor functions (Address Resolution, ...), it also offers new functions, like *Router Discovery* and *Stateless Address Autoconfiguration*. However, the NDP initially offers no protection mechanism and is prone to address spoofing and Denial of Service attacks. To enhance the security of the NDP, the IETF defined a security extension: Secure Neighbor Discovery [2] (SEND). This extension relies on particular IPv6 addresses named Cryptographically Generated Addresses [3] (CGA), a form of Cryptographically Based Identifier derived from the Statistically Unique Cryptographically Verifiable Addresses [4]. Basically, CGA addresses are IPv6 addresses where the *Interface Identifier* part (i.e. the 64 rightmost bits of the address) is a hash of a public key and other public data of a node. With SEND extensions, the node can later prove CGA address ownership by signing messages with its private key, thus mitigating spoofing attacks (other mechanisms also ensure protection against replay attacks). However, SEND's protection is a double-edged sword: while it offers protection to the node, it prevents functions that usually require a third party node to modify or emit NDP messages and it disables any form of address sharing. To be more precise, it breaks two functions: the Proxy Neighbor Discovery (Proxy ND), used by [5], [1] and [6], and the IPv6 anycast addresses, defined in [1] and recently used in [6]. Since these functions can not rely on SEND to protect the address and authenticate the routers, they are vulnerable to the attacks targeting the NDP that are listed in [7].

Proposals to extend the SEND protocol to allow compatibility with the Proxy Neighbor Discovery function are proposed in documents [8], [9], [10], [11]. Among them, documents [10] and [11] solve IPv6 anycast addresses sharing problem. Currently, preferred solution at the IETF [11] relies on a specific authorization granted to some nodes, routers or *proxies*, to bypass the CGA protections, thus empowering a small subset of nodes and turning them into a privileged target to attackers. We propose in this document an alternate mechanism that we think surpass the other proposals in term of simplicity, flexibility and integration.

We advocate that on-going efforts on the Signature Algorithm Agility for SEND [12] (and its companion document [13]) could be extended to provide solutions on the two aforementioned issues. This work removes SEND and CGA's limitations in term of Signature Algorithms. In other words, it allows a node to select among different Signature Algorithms and it extends the CGA addresses to support multiple public keys of different types (e.g. RSA, ECC) for the same node. Such CGA addresses are referred to as *Multiple-Key CGA* or (MCGA). In this document, our contribution is threefold. Firstly, we propose modifications to the CGA addresses and

the SEND protocol to support Signature Algorithm Agility and present the MCGA addresses. Secondly, we extend the MCGA addresses to store public keys of different nodes, enable a secure address sharing and to solve incompatibilities between address sharing functionality of the Proxy ND and the address protection of the SEND protocol. Finally, we further extend MCGA to secure the Anycast address sharing and enable the SEND protection in these environments.

As a brief overview, we introduce the Neighbor Discovery Protocol, the Cryptographically Generated Addresses and the Secure Neighbor Discovery Protocol (Sec. II). Then, we present our Signature Algorithm Agility extensions (Sec III). Subsequently, we describe the different variants of Proxy ND (Sec. IV-A and Sec. IV-B) and the incompatibilities between SEND security model and the Proxy ND (Sec. IV-C). We present a brief overview of the existing Secure Proxy ND solutions and analyse their weaknesses (Sec. IV-D). We propose a novel Secure Proxy ND solution based on Signature Algorithm Agility (Sec. V) and we study its limitations (Sec. V-A). We then present our last contribution and focus on defining a secure IPv6 Anycast addressing proposal. We introduce the incentives for this work and present the similar works (Sec. VI). We then describe a new solution based on our Signature Algorithm Agility proposal (Sec. VI-E). At last, we present an implementation of our proposal and evaluate its performances (Sec. VII). Finally, we conclude with remaining open issues (Sec. VIII).

II. Securing the Neighbor Discovery Protocol

A. Neighbor Discovery Protocol

The Neighbor Discovery Protocol (NDP) [1] provides functions for nodes that are located on a same link (i.e. neighbors). Among these functions, we can list the *Address Resolution*, that maps IPv6 addresses to Link-Layer addresses, the *Prefixes Discovery* and *Routers Discovery*, that respectively discover new prefixes and routers (i.e. IPv6 nodes that are not "hosts") over a link. Document [14] extends the NDP and introduces the *Stateless Address Autoconfiguration*, i.e. a procedure that relies on the *Prefix Discovery* to enable hosts to build their own IPv6 addresses.

A set of five ICMPv6 messages implements the NDP and Stateless Address Autoconfiguration functions:

- *Neighbor Solicitation* (NS) and *Neighbor Advertisement* (NA) messages enable a node to respectively request and answer a neighboring node for its *Link-Layer* address (as part of the Address Resolution function) ;
- *Router Solicitation* (RS) and *Router Advertisement* (RA) messages enable a node to query routers for on-link informations, such as the *Valid Prefix* or the *Maximum Transfer Unit* (MTU) ;
- Routers use *Redirect* messages when nodes communicate through a router whereas they could in fact communicate directly (i.e. they are neighbors) ;

B. Cryptographically Generated Addresses

Cryptographically Generated Addresses [15], [3] (CGA) are a specific type of IPv6 Unicast addresses that binds a public key to the *Interface Identifier* part of an address. The CGA generation algorithm consists in computing two specific hashes, *Hash1* and *Hash2*, over a specialized data structure named *CGA Parameters* (illustrated in Figure 1) which contains node's *Public Key*, the *Subnet Prefix*, an optional *Extension* field (i.e. for future CGA extensions) and other information. The *Hash2* calculus strengthens resistance of the CGA address to brute force attack by artificially raising computational costs of the address generation procedure. This technique is referred as "hash extension" in [15]. *Hash1* contains the 64 leftmost bits of a SHA-1 digest computed over the *CGA Parameters*. After the 3 leftmost bits are set to the 3 bits of the encoded SEC value and the *U* and *G* bits (detailed in [16]) are set to 0, these 64 bits form the CGA's *Interface Identifier*. The *Interface Identifier* is then concatenated to the *Subnet Prefix* (learnt during the *Routers Discovery* procedure) to form the CGA address.

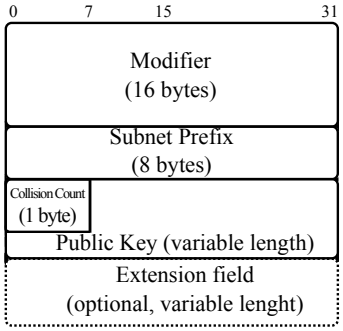

Fig. 1. Format of the *CGA Parameters* data structure

CGA verification process consists in computing the hash values and doing comparison with the *Interface Identifier* part of the address. This implies, as we will see Section II-C, that the *CGA Parameters* are transfered to neighboring nodes prior to the verification of the CGA address.

C. Secure Neighbor Discovery Protocol

While CGA's purpose is to bind a public key to an IPv6 address, the Secure Neighbor Discovery [2] (SEND) provides the address ownership and ensures message authenticity, integrity and freshness. SEND protection is twofold: it protects the node from address spoofing and provides to the host a mechanism to authenticate its Access Router.

SEND provides four new options to NDP messages:

- *CGA Option* encapsulates the *CGA Parameters* in a NDP message. It is needed for the CGA verification procedure ;
- *RSA Signature Option* (illustrated Figure 2) includes the *Digital Signature* performed with the private key linked to the CGA address and thus it proves the address ownership. It ensures authenticity and integrity of the ICMPv6 header and the NDP options ;

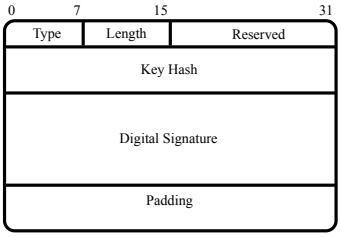

Fig. 2. Format of the *RSA Signature Option*

- *Nonce Option* and *Timestamp Option* provides anti-replay protection ;

In addition, a new function, named *Authorization Delegation Discovery* (ADD) defines two new ICMPv6 messages, *Certificate Path Solicitation* (CPS) and *Certificate Path Advertisement* (CPA). This procedure enables hosts to verify the authenticity of the neighboring routers and thus their RA messages. Basically, in order to authenticate neighboring routers, a host sends a CPS message and receives CPA messages from the routers. The CPA messages form a Certificate Path from a Trust Anchor to the router's certificate. The host can later check that the key pair used to build the CGA address and the *RSA Signature Option* matches the public key contained in the certificate.

III. Signature Algorithm Agility

CGA and SEND as currently defined in [2], [3] are linked to the SHA-1 hash function and to the RSA Signature Algorithm. Recently, due to progress in SHA-1 cryptanalysis [17] and rise of interest in Elliptic Curve Cryptography (ECC), the IETF defined new work items to provide CGA and SEND the ability to select new cryptographic algorithms. A first document [18] offers support for different hash algorithms in CGA generation and verification. However, the SEND protocol currently remains dependent on SHA-1 and RSA to provide Digital Signatures. In order to fill the gap, we propose a Signature Algorithm Agility support for SEND and we define two extensions to the protocol. We also submitted these extensions to the IETF [13], [12].

First, we improve CGA with the *Multiple-Key CGA* (MCGA). MCGA extends CGA addresses and allows them to carry and to be built from more than one public key, as illustrated in Figure 3. In practice, each extra public key is stored in a *CGA Parameters* field named *Public Key* extension. This enables a node owning a CGA address to sign (possibly multiple times) SEND messages with any of the different *Private Keys* associated to the CGA address. In the Signature Agility context, it enables nodes to negotiate algorithms in heterogeneous networks (i.e. between nodes with different set of signature algorithms) and securely perform *Neighbor Discovery* (ND) functions. The extra *Public Keys* are stored in the extension field of the *CGA Parameters*.

Secondly, we modify the *RSA Signature Option*. The purpose of this extension is twofold: a node is now able to point out the Signature Algorithm that was used to construct

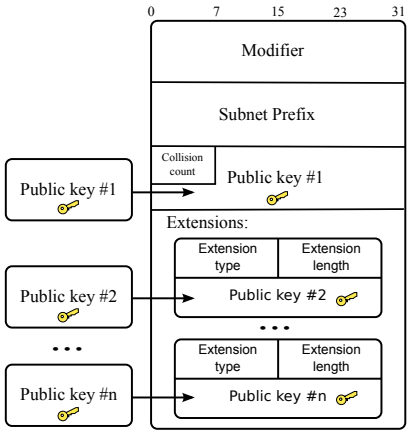

Fig. 3. Localisation of the *Public Key* extensions within the *CGA Parameters* data structure

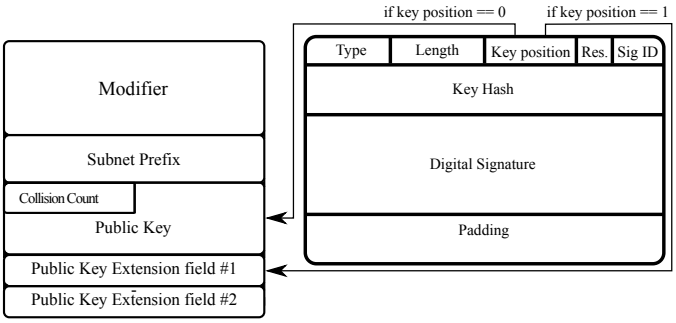

Fig. 4. Public key selection mechanism embedded within the *Universal Signature Option*

the *Digital Signature* (e.g. Elliptic Curve Digital Signature Algorithm, ECDSA) along with the associated Public Key within the *CGA Parameters* (see Figure 4). This modification is named *Universal Signature Option* (USO).

IV. Proxy Neighbor Discovery

In this document, we use "Proxy Neighbor Discovery" (Proxy ND) as a generic term that refers to the function offered by a third party node, the *proxy*, when it advertises or modifies ND messages on behalf of a neighboring node. Such neighboring node is called "protected node" whereas the corresponding address(es) is referred to as "protected address(es)".

Proxy ND refers to two distinct scenarios described in subsections IV-A and IV-B.

A. Proxy ND definition in RFC 4861 & RFC 3775

When the NDP [1] was initially defined, it incorporated a mechanism to protect nodes that are located on a different link of a proxy or accessible via a tunnel. This mechanism, the Proxy ND, was later refined in the Mobile IPv6 specification [5] where a Mobile Node requires assistance of the proxy to protect its *Home Address*(*es*) when moving away from the *Home Network*. In practice, the proxy transmits spontaneous *Neighbor Advertisement* or responds to *Neighbor Solicitation*

messages addressed to the protected address, and adds its Link-Layer Address as the *Source Link-Layer Address* option or *Target Link-Layer Address* option.

B. Proxy ND definition in RFC 4389 (ND Proxies)

The need for "ND Proxies", defined in [6], originates from specific networks where a single IPv6 subnet spans among multiple links and where nodes from these multiple links must communicate. In this context, the proxy relays NDP messages from one link to others and modifies on the fly the content of the *Source Link-Layer Address* option and the *Target Link-Layer Address* option to place its own Link-Layer Address (see Fig. 5). Thus, nodes situated on different links only learn the proxy's Link-Layer address (e.g. during *Address Resolution* procedure) and then the following traffic between nodes is forwarded through the proxy.

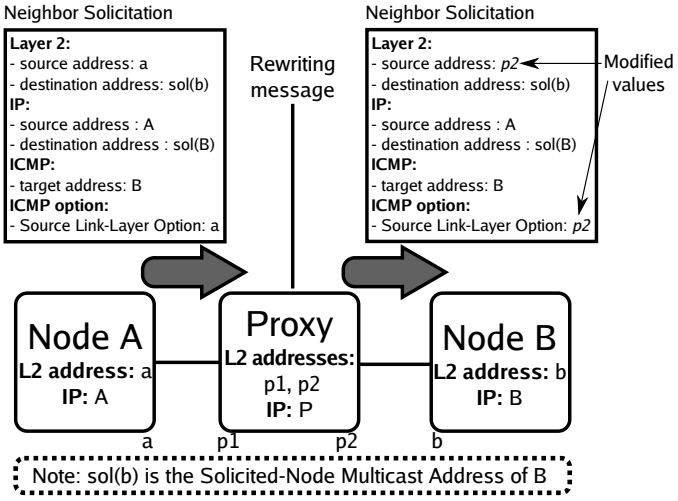

Fig. 5. A RFC 4389 proxy modifies a Neighbor Solicitation message passing through

This is particularly useful to offer bridge-like function between links of different Link-Layer technologies and to prevent the use of Network Address Translation (NAT) when a single subnet must be shared among several links.

C. Incompatibilities between Proxy ND and SEND

Section IV-A and Section IV-B presented two scenarios where the proxy respectively (1) emits packets on behalf of the protected nodes and (2) modifies messages. With the CGA verification and the RSA Signature option protection, SEND [2] ensures that only the owner of the address is enabled to emit message with its source address and that the message's integrity is valid. As such, SEND and Proxy ND are incompatible for both scenarios (1) and (2).

D. Works related to Secure Proxy ND

The literature already provides some documents that propose a solution for a Secure Proxy Neighbor Discovery. However, we identified limitations for each solution that make them unsuitable for some usages.

Among them, Krishnan et al. present in [11] a certificate-based solution (currently in standardisation process at the IETF). The router's certificate is extended to support a new *Extended Key Usage*[1] (EKU) field that indicates whether the router also assumes a *proxy* role for this subnet or not. During the *Authorization Delegation Discovery* (ADD), as part of SEND, neighboring nodes learn that the router is also authorized to act as a proxy for subnet prefix it advertises. Then, whenever the proxy issues or modifies ND messages and signs with its public key, the neighboring nodes will trust the message. While this solution is pretty easy to deploy (i.e. slight modifications in the certificates), we advocate that it grants too much power to the proxy entity and that an attacker could exploit this central position.

Another certificate-based solution, proposed by Nikander and Arkko, described in [8] and [19], empowers the nodes to determine if a router is trusted enough to be a proxy and to issue a certificate to authorize it to act as such. However, the proposal fails to identify the real overhead due to the certificate exchange mechanism.

In document [10], Kempf et al. proposes a smart solution, based on a Ring Signature scheme: the Rivest-Shamir-Tauman (RST) algorithm. In simple words, the Ring Signature is a cryptographic mechanism, similar to the Group Signature, which supports anonymity of the signer (i.e. signer's signature is indistinguishable from the signature performed by the other members of the ring). In this solution, the CGA address is formed from the public keys of the different members of the ring. The Ring Signature makes it impossible to distinguish between the *protected node*'s signature and the *proxy*'s one. The authors claim their solution is especially important in IP Mobility solution such as [5] where an attacker is prevented to know when the node left. However, while the signer's identity remains hidden, the paper does not explain why an attacker could not monitor changes in *Source Link-Layer Address* option and *Target Link-Layer Address* option to identify the real emitter. Indeed, both options contains the sender's Link-Layer address (e.g. MAC address) and reveal the identity of the sender. Consequently, we believe that this solution cannot offer anonymity function. Another limitation is that the *CGA Parameters* data structure and the *Digital Signature* sizes are linear functions which depends on the number of entities participating in the Ring Signature.

For the sake of completeness, we note the existence of a proposal by Haddad and Naslund, presented in [9]. This solution, still at an early stage, weakens the CGA protection mechanism and thus should not be considered as a viable solution.

To the best of our knowledge, only the solution [10] has been implemented.

[1]The EKU extension is an X.509 certificate extension that restricts the usage of a certificate to a specific usage.

V. Securing Proxy ND using Signature Algorithm Agility

In this paper, we present a solution that enables the protected node to decide whether to trust the proxy and to authorize proxy's function accordingly. Regarding this aspect, our solution is really close to solution depicted in [19]. On a practical point of view, we propose to use MCGA and the *CGA Parameters Public Key* extension fields to store multiple public keys which do not belong to the same node. More precisely, we propose to build a MCGA based on the node's public key and the (potentially multiple) proxys' public keys. Upon joining the network, prior to the generation of any CGA addresses, a node is requested to send a *Router Solicitation* message using the *unspecified* address as source address. When receiving the message, the routers respond with *Router Advertisement* messages and trigger node's *Authorization Delegation Discovery* when their public key is unknown to the node. During this exchange, the node receives neighboring Routers' public keys and certificates. Similarly to [11], the node checks the certificate for the EKU extensions indicating the *proxy* role. Whenever a router is authorized to act as a proxy ND, the node stores its Public Key in the extensions field of its *CGA Parameters* data structure. Thus, after the node acquired proxys' public key, the MCGA can be built. Shortly thereafter, as part of the *Duplicate Address Detection* procedure, the node sends *Neighbor Solicitation* messages. From these messages, proxys learn the new MCGA addresses and the content of the *CGA Parameters*. Afterwards, any of the Public Keys stored in the *CGA Parameters* can interchangeably be used to sign the ND message or, said in other words, the address ownership is shared between the node and the proxy(s).

Note that this "*Proxy Discovery*" procedure does not introduce extra delays during the *Stateless Address Autoconfiguration* as the node is usually required to wait for the *Prefix Discovery* procedure to complete prior to the construction of new addresses.

A. Limitation of the solution

Our solution relies on the MCGA to store several public keys. This implies that SEND secured NDP messages embed the *CGA Option* containing multiple *Public Keys* extensions. Hence the size of the message grows with the number of public keys. Nonetheless, contrarily to solution [10], the size of the Digital Signature remains fixed. However, it implies an upper bound to the number of Public Keys that can be stored in *CGA Parameters*. We estimated this upper bound to be eleven public keys when using 256-bit ECC keys. More public keys could require the fragmentation of ND messages, with its side effects, as it will exceed IPv6 minimal *Maximum Transfer Unit* (MTU) value (1280 octets). However, we are confident that, in real world scenarios, the nodes will not be required to simultaneously authorize ten proxys at once.

Another drawback is that the security level of a MCGA depends on the security level of the node's public key and the security level of the proxy(s)'s public key(s). The weakest public key determines the overall strength of the MCGA. We advise that, when possible, all the nodes of the same administrative domain should be deployed with equally secured type and length of public keys. In a global fashion, a node must reject proxy's public keys having a lower security level than its own.

Finally, unlike the solution presented in [11] or [19], we can not add nor remove proxy after the CGA address is generated. This limitation is especially harmful for long lived connections when news proxys arrive on the network.

VI. Anycast Addressing

A. Link-Local Anycast

This section focuses on Link-Local Anycast where nodes sharing a link uses the same IPv6 address, as illustrated in Figure 6. This is a different approach than Anycast Routing (e.g. provided by the Border Gateway Protocol, BGP). From

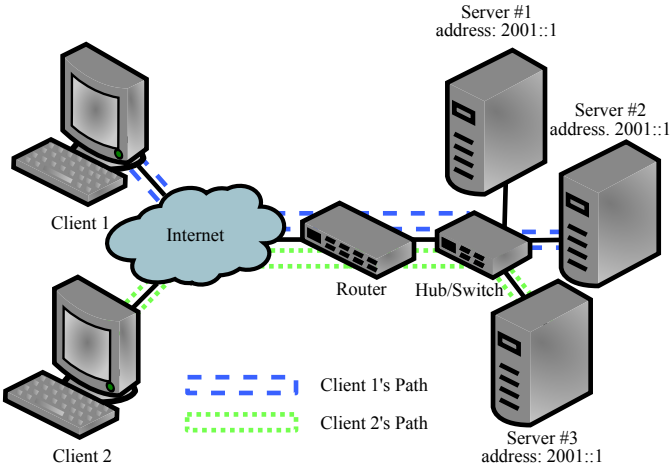

Fig. 6. Link-Local Anycast topology example

a correspondent node perspective, IPv6 Anycast addresses are undistinguishable from IPv6 Unicast addresses. However, from a practical standpoint, IPv6 Anycast slightly deviates from IPv6 Unicast addresses' behavior. Namely, upon receipt of a *Neighbor Solicitation* towards an IPv6 anycast address (e.g. as part of the *Address Resolution* procedure), the nodes sharing the IPv6 address wait during a random delay before sending a *Neighbor Advertisement* (NA) message. This random delay decreases the probability of a congestion and offers a simple load-balancing mechanism to spread the load among the nodes. Also, the NA contains an *Override Flag* which indicates if previously received informations should be updated. In NA messages sent from anycast addresses, this flag must be off, meaning that a new NA must not update a previous one. As a result, this rule ensures that, once a communication is established, the correspondent node will never get redirected to another node sharing the address.

B. Proxy Mobile IPv6 (RFC 5213)

Proxy Mobile IPv6, defined in [20], proposes a network-based local mobility management. The main purpose of this

protocol is that no modification is required on the "Mobile" Node (MN) so that it only needs to implement a basic IPv6 stack. As a result, all the complexity involved in mobility management is handled on the network side. Two new entities, the *Mobile Access Gateway* (MAG) and the *Local Mobility Anchor* (LMA) are defined. The MAG ensures the role of Access Router. In order to avoid possible Link-Local address collisions between the MN and the different MAGs, it is required that when the MN first joins the network, the LMA creates a mobility session for the MN and determines a Link-Local address shared among the different MAGs. Afterwards, upon node movement, the new MAGs (i.e. MAG to which the MN establishes a new connection with) query the LMA so they offer the same Link-Local address to the MN.

C. Incompatibilities between Anycast addresses and SEND

Section VI-A and Section VI-B present two mechanisms relying on IPv6 address sharing. However, SEND only authorizes the owner of the address, i.e. the owner of the public/private key pair, to send NDP signaling. It would require nodes willing to share an address to use the same key pair among the nodes. Nonetheless, it would weaken the security level of the solution and should not be considered as a solution. Therefore, it leads to an incompatibility between the two functions.

D. Works related to secure Anycast addresses

To the best of our knowledge, only citeringsig and [11], presented in Section IV-D, offer the ability to use anycast addresses in SEND protected environment. [11] restricts the address sharing to router nodes (authorized through their certificates to advertise any address, thus allowing them to share identical ones). This limitation is not present in [10] that allow multiple nodes to share an address by forming a Ring and building a CGA address from their public keys.

E. Using Multiple-Key CGA to store Neighbor's Public Key

Similarly to the solution in Section V, we propose to store the public keys of the multiple nodes sharing the address in the *Public Key extension* fields of the *CGA Parameters*. Thus, the anycast address is formed from multiple public keys and each node sharing the address possesses the private key associated to one of these public keys. The *Universal Signature option* then enables nodes to point the public key they use to generate Digital Signatures and to bind their private key accordingly.

The main difficulty lies in securely sharing the Public Keys and the other *CGA Parameters* between nodes. For MAG nodes in the PMIPv6 scenario (Sec. VI-B), this function could be ensured by the LMA. However, for Link-Local Anycast scenario (Sec. VI-A), at this stage of our solution, key distribution is not automated. One simple solution would be to assume the presence of an administrator (i.e. a trusted party) that manually distributes the public keys and the specific private key on each node sharing an address. Of course, this solution is a time consuming process. For this reason, we propose an alternate (and automated) solution that allows nodes to have a "learning phase" during which they learn their neighbor's public keys. During this phase, the nodes trust every protected NDP message they receive. Once the learning phase is over, the nodes then build their Multiple-Key CGA and start sharing the address. This proposal implies that only trustworthy nodes are present during the "learning phase". It is a viable condition during the deployment of a new network. We also note that the "learning phase" does not necessarily require new message as *Neighbor Advertisement* message could be used to broadcast the public key.

Due to the similarities with the solution described in Section V, the limitations given in Section V-A apply to this section as well.

# of Keys	1	2	4	11
Message size	312	408	592	1232
CGA generation	1.142	1.173	1.204	1.425
Sig. generation	0.075	0.077	0.080	0.097
Sig. verification	0.035	0.036	0.037	0.045

TABLE I
PERFORMANCE ANALYSIS OF SEVERAL OPERATIONS ON A NEIGHBOR SOLICITATION MESSAGE (TIMES ARE IN SECONDS, SIZES ARE IN BYTES)

VII. PERFORMANCE ANALYSIS

As a preliminary work, we extended our NDprotector[2] project to support MCGA and the Signature Algorithm Agility. NDprotector is a highly customisable CGA & SEND implementation exclusively written in Python. This implementation privileges the protocol correctness over the performances. As such, NDprotector is slower than some other more optimized implementations (such as the one developed by NTT DoCoMo[3]). Nonetheless, the time measurements are sufficient to assess the overhead when the number of *Public Keys* extensions increases in the MCGA.

In order to prove our solution's feasibility, we implemented and tested our proposal to secure anycast addresses (presented in Sec. VI-E).

Table I illustrates the performances of our proposal during the CGA address generation, *Neighbor Solicitation* signature generation and verification. We consider 10000 samples for each Public Key storage configuration, as was already considered in previous works on CGA performance measurements [21]. Also based on our past experience, we preferred ECC keys over RSA keys due to their advantages regarding signature size, key size and speed. To be more precise, we decided to use 256-bit ECC keys (equivalent to 2048-bit RSA keys in term of security level).

Figure 7 shows that the number of public keys contained in the MCGA, as expected, does not significantly influence the signature generation and verification duration. However, as the number of public keys stored in the MCGA increases, the

[2]http://amnesiak.org/NDprotector

[3]NTT DoCoMo stopped support for their implementation. However, some people of the MobiSEND project maintain their code: http://mobisend.org/software.html

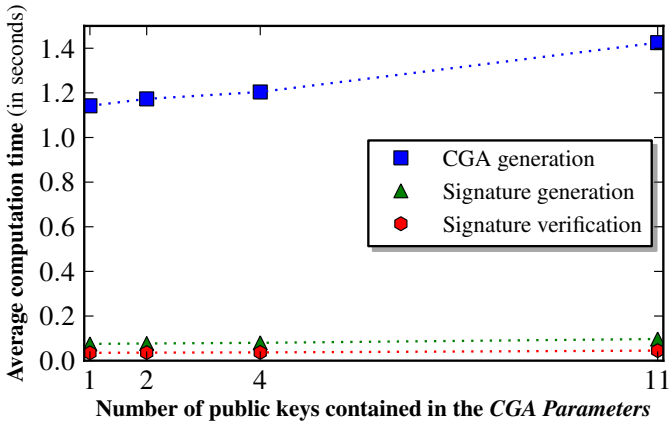

Fig. 7. Graphical representation of performance analysis of several operations on a *Neighbor Solicitation* message

computation time of CGA generation (i.e. *Hash1* and *Hash2* computation) and the message size grow linearly[4].

In the worst case, when an anycast address is shared between eleven nodes, the CGA generation time increases by 25% compared to a CGA address containing a single public key. This percentage is excessive at a first glance, but it should be counterbalanced by the fact the CGA generation is only performed once, during the node initialisation.

Among the different solutions presented in IV-D, only the Ring Signature based proposal [10] has been implemented and provides performance results. The authors measured the signature generation and verification speed and the digital signature size. Unfortunately, their implementation is not publicly available and hence, we can not recompute the signature generation and verification duration within our testbed. Consequently, we can not confront their performance results with ours. However, we can compare the signature size, which do not require us to have the same testbed setup. In the document [10], the digital signature grows with the number of nodes sharing the address (from 300 bytes to 1500 bytes), whereas in our solution, the digital signature remains constant (71 bytes) regardless the number of nodes. We believe that the small size of the signature in our messages is a huge advantage over the solution presented in [10], especially in mobile scenarios, where wireless bandwidth is a scarce resource.

VIII. Conclusion

In this document, we identify the issue of address sharing for SEND protected nodes. Our presentation of the existing solutions reveals that they suffer at best from mild limitations. We then propose a novel solution, based on Signature Algorithm Agility, that offers, at almost no cost, a solution to secure Proxy ND and anycast addresses. We provide a fair analysis of the limitations of our solution. We demonstrate through an implementation that this mechanism is realistic. The performance analysis shows that our proposal does not introduce a large overhead and offers advantages over existing solutions [10].

In our future works, we plan to define a negotiation mechanism, so that anycast addresses creation could be performed with almost no involvement of any operator. Additionally, we plan to extend the NDprotector implementation to support our Secure Proxy ND proposal.

[4]bumps in Fig. 7 are due to SHA-1 hash function, which process data by blocks of 512 bits

Acknowledgment

This research is part of the TLCOM MobiSEND project financed by the French National Research Agency (ANR). Tony Cheneau is supported by a grant of TELECOM SudParis.

References

[1] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," IETF, RFC 4861, Sep. 2007.
[2] J. Arkko, J. Kempf, B. Zill, and P. Nikander, "SEcure Neighbor Discovery (SEND)," IETF, RFC 3971, Mar. 2005.
[3] T. Aura, "Cryptographically Generated Addresses (CGA)," IETF, RFC 3972, Mar. 2005.
[4] G. Montenegro and C. Castelluccia, "Statistically Unique and Cryptographically Verifiable (SUCV) Identifier and Addresses," in *9th Annual Network and Distributed System Security Symposium (NDSS)*, Feb. 2002.
[5] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6," IETF, RFC 3775, Jun. 2004.
[6] D. Thaler, M. Talwar, and C. Patel, "Neighbor Discovery Proxies (ND Proxy)," IETF, RFC 4389, Apr. 2006.
[7] P. Nikander, J. Kempf, and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats," IETF, RFC 3756, May 2004.
[8] J.-M. Combes, S. Krishnan, and G. Daley, "Securing Neighbor Discovery Proxy: Problem Statement," IETF, RFC 5909, Jul. 2010.
[9] W. Haddad and M. Naslund, "On Secure Neighbor Discovery Proxying Using 'Symbiotic' Relationship," IETF, Internet-Draft draft-haddad-csi-symbiotic-sendproxy-01, Jul. 2009.
[10] J. Kempf, J. Wood, Z. Ramzan, and C. Gentry, " IP Address Authorization for Secure Address Proxying Using Multi-key CGAs and Ring Signatures ," in *Advances in Information and Computer Security*. Springer, 2006, pp. 196–211.
[11] S. Krishnan, J. Laganier, M. Bonola, and A. Garcia-Martinez, "Secure Proxy ND Support for SEND," IETF, Internet-Draft draft-ietf-csi-proxy-send-05, Sep. 2010.
[12] T. Cheneau, M. Laurent-Maknavicius, S. Shen, and M. Vanderveen, "Support for Multiple Signature Algorithms in CryptographicallyGenerated Addresses (CGAs)," IETF, Internet-Draft draft-cheneau-csi-cga-pk-agility-00, Oct. 2009.
[13] T. Cheneau, M. Laurent, S. Shen, and M. MichaelaVanderveen, "Signature Algorithm Agility in the Secure Neighbor Discovery (SEND)Protocol," IETF, Internet-Draft draft-cheneau-csi-send-sig-agility-00, Oct. 2009.
[14] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," IETF, RFC 4862, Sep. 2007.
[15] T. Aura, "Cryptographically Generated Addresses (CGA)," in *Information Security*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003, vol. 2851, pp. 29–43.
[16] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture," IETF, RFC 4291, Feb. 2006.
[17] C. D. Cannière and C. Rechberger, "Finding SHA-1 Characteristics: General Results and Applications." in *ASIACRYPT*, 2006, pp. 1–20.
[18] M. Bagnulo and J. Arkko, "Support for Multiple Hash Algorithms in Cryptographically Generated Addresses (CGAs)," IETF, RFC 4982, Jul. 2007.
[19] P. Nikander and J. Arkko, "Delegation of signalling rights," in *Security Protocols Workshop*, ser. Lecture Notes in Computer Science, vol. 2845. Springer, 2002, pp. 203–214.
[20] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, "Proxy Mobile IPv6," IETF, RFC 5213, Aug. 2008.
[21] 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.