CGA & SEND maintenance	T. Cheneau	
Internet-Draft	M. Maknavicius	
Updates: RFC3971	TMSP	
(if approved)	S. Shen	
Expires: December 7, 2009	Huawei	
	M. Vanderveen	
	Qualcomm	
	June 05, 2009	

TOC

Signature Algorithm Agility in the Secure Neighbor Discovery (SEND) Protocol
draft-cheneau-send-sig-agility-01

Status of this Memo

Copyright Notice

Abstract

This draft describes a mechanism to enable the Secure Neighbor Discovery (SEND) protocol to select between different signature

algorithms to use with Cryptographically Generated Addresses (CGA). It also provides optional support for interoperability between nodes that do not share any common signature algorithms.

Table of Contents

---

1. Introduction

TOC

The usage scenarios associated with neighbor discovery have recently been extended to include environments with mobile or nomadic nodes. Many of these nodes have limited battery power and computing resources. Therefore, heavy public key signing algorithms like RSA are not feasible to support on such constrained nodes. Fortunately, more

lightweight yet secure signing algorithms do exist and have been standardized, e.g. Elliptic Curve based algorithms.
It is then a worthwhile goal to extend secure neighbor discovery to support signing and corresponding hashing algorithm agility. Besides accommodating power-constrained nodes, signing and hashing algorithm agility is also desired as a safety measure over time, to offer alternatives when cryptanalysis of one type of algorithm makes significant progress.
The aim of this memo is to outline options for allowing public key signing algorithm and hashing algorithm agility for nodes configured to perform secure neighbor discovery operations. The extent to which these options impact existing specifications [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.) and [RFC3972] (Aura, T., "Cryptographically Generated Addresses (CGA)," March 2005.) is also addressed.

2. Overview

TOC

2.1. Compatibility with existing specifications

TOC

The current SEND protocol specification, [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.), mandates the use of the RSA signature algorithm. Since the time of its writing, different signature algorithms have been shown to be secure and have been adopted by other protocols in an effort to reduce key length, signature generation and verification time, and increase security level. This shift in signature algorithm adoption particularly benefits lightweight devices, which are power and memory-limited but in need of secure signing algorithms support. For these reasons, we feel that the restriction on the signature algorithm for SEND is no longer warranted.

2.1.1. Classification of SEND nodes

TOC

At the time of this writing, there are no known large-scale or even small-scale deployments of [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.)-compatible devices. However, in the interest of caution, we assume that there exist nodes that support only the RSA algorithm and that are configured to perform secure neighbor discovery. Such nodes may not be

updated in the near term or for the foreseeable future. On the other hand, it appears that there will be deployments of nodes that support only Elliptic Curve Cryptography as their public key algorithm, i.e. ECDSA as a signature algorithm, rather than traditional RSA.
To ensure that all possible network/link configurations are considered when designing a signature agility solution, we categorize nodes (hosts and routers) according to their support for different signature algorithms, as follows:

Type H1 host:
A host that only supports one type of signature algorithm and has a CGA generated with the public key of this algorithm.

Examples of this type of hosts: an old host that does not support signature agility, i.e. only supports RSA signature algorithm; or, a host that only supports ECDSA signature.

Type H2 host:
A host that supports multiple signature algorithms and has a CGA generated with only one key selected from among its supported algorithms.

Examples of this type of hosts: (1) a host that supports RSA and ECDSA signature algorithms, but only has a CGA derived with an RSA public key; (2) a host that supports RSA and ECDSA signature algorithms, but only has a CGA derived with an ECC public key.

Type H3 host:
A host that supports multiple signature algorithms and has a CGA generated with multiple keys of different supported algorithms.

Such CGA generation is made possible by the introduction of a new CGA extension (see companion draft [cheneau-cga-pk-agility] (Cheneau, T., Laurent-Maknavicius, M., Shen, S., and M. Vanderveen, “Support for Multiple Signature Algorithms in Cryptographically Generated Addresses (CGAs),” June 2009.)). Such hosts can be compatible with hosts of other types for secure neighbor discovery.

Type H4 host:
A host that supports multiple signature algorithms and has multiple CGAs, each of which is associated with a single key of one supported algorithm. For simplicity, we do not consider hosts that have multiple CGAs, one or more of which are generated from multiple public keys.

A node MUST select and settle on one CGA when building a trust relationship with another device via SeND (more below). In such

> cases, a destination node may be reached at a CGA associated with a signature algorithm that the originating node cannot verify. The destination node will need to securely redirect the originating node to one of its other CGA(s) (presumably with a common signature algorithm). The need for a method to secure the binding between the two CGAs of the destination node is still an open problem.
>
> Based on this reasoning, consideration of H4 type nodes is left for future work.

Routers are more likely to possess the resources necessary to support multiple signature and hashing algorithms. It is also more feasible that routers employ certificates. However, for a basic signature agility solution, we do not mandate that routers support multiple signature and hashing algorithms.
Possible router devices with different signature algorithm support ability are:

> **Type R1 router:**
> A router that only supports one type of signature algorithm and has a CGA and Certificate with a public key of this algorithm.
>
> Such routers are expected to be commonplace, as compliance with [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.) suffices for them.
>
> **Type R2 router:**
> A router that supports multiple types of signature algorithms and has one CGA and Certificate with a public key of one of the algorithm types.
>
> This type of router can sign and verify signatures of the type of certificate it owns, and additionally, it can verify signatures of other algorithm types.
>
> **Type R3 router:**
> A router that supports multiple types of signature algorithms and has one CGA composed of multiple Publics Keys and multiple certificates containing each a Public Key.
>
> **Type R4 router:**
> A router that supports multiple types of signature algorithms and has multiple CGAs and Certificates with public key of several different algorithm types.
>
> This type of router can sign and verify signatures of multiple types. Such routers may not be attractive to build and deploy due

> to increased requirements on its resources. Moreover using multiple CGAs (with no bindings) may make that routers appear as having multiple identities.

Note that all types of router presented above can be configured to use SEND over multiple interfaces or to have multiple addresses on the same interface. In this case, the router will use separate CGAs. Such configuration is treated in this draft as if the different addresses refer to separate entities.

2.1.2. Principal Scenarios

TOC

Based on the discussion above, a SEND agility solution should at least properly deal with the communication between devices of type H1, H2, H3, R1, R2 and R3.

> An H1 or R1 node interacting with an H2 or R2 node: i.e., a node supporting only RSA (for example, an old non-agility node which only supports RFC3971) and a node supporting both RSA and ECDSA (or other new algorithms). These two nodes may be able to perform secure neighbor discovery.
>
> An H1 or R1 node interacting with another H1 or R1 node, but their algorithms differ: e.g., a node supporting only RSA (for example, an old non-agility node which only supports RFC3971) and a node supporting only ECDSA (or other new algorithms). In this case, implementations supporting SEND signature agility solution may likely realize the incompatibility, while older implementations may not.
>
> A node of any type (H1, H2, H3, R1, R2, R3) interacting with another node, their algorithms differ but there is a 3rd party willing/able to help: this is an optional solution applicable to the previous scenario, where two nodes that support SEND but do not have any signature algorithms in common can talk through a third party (router). In this case they should be able to perform facilitated secure neighbor discovery.
>
> An H2, H3 or R2 node interacting with another H2, H3, or R2 node: e.g., two nodes that support at least one signature algorithms in common will be able to perform secure neighbor discovery.
>
> An additional rule for H2, H3 or R2, R3 node interacting with another H2, H3, or R2, R3 node applies: two nodes that support two or more signature algorithms in common (one of which is likely preferred over the other), will be able to perform secure neighbor discovery with any of these signature algorithms.

2.2. Agility Requirements

TOC

We hold the following to be requirements on a signing algorithm agility solution for SEND:

- *A Signature-Algorithm-Agility-Node should be able to communicate with a Non-Signature-Algorithm-Agility-Node, but not necessarily employ SEND. Traditional ND should suffice, to accommodate nodes that only support one type of Signature Algorithm, which may not be RSA. Local policy MAY disable this behavior, namely the use of unsecured ND messages when communicating with a node that does not share any common signature algorithm.

- *Two Signature-Algorithm-Agility nodes that support a common Signature Algorithm and hashing algorithm should be able to communicate using SEND and sign messages using the common Signature Algorithm and hash algorithm.

- *The current SEND/CGA specifications should incur as few changes as possible.

2.3. Mechanism for Agility Support of CGA and SEND

TOC

To achieve signature agility for SEND, it must be possible for a CGA to be generated from and to be securely associated with multiple public keys corresponding to different signature algorithms. This capability is described in the companion draft [cheneau-cga-pk-agility] (Cheneau, T., Laurent-Maknavicius, M., Shen, S., and M. Vanderveen, “Support for Multiple Signature Algorithms in Cryptographically Generated Addresses (CGAs),” June 2009.).
This document proposes an update to [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.) to allow two SEND nodes to choose an appropriate signature algorithm. This solution encompasses the following:

- *A "Supported Signature Algorithm" Neighbor Discovery Protocol option which contains a list of signing and hashing algorithms that the sender node supports for SEND purposes and its interaction with the Neighbor Cache;

- *A modification of the "RSA Signature" option defined in the SEND specification;

*An optional solution to support secure communication through a router acting as a third party when nodes don't share any common Signature Algorithm.

We define the aforementioned options format and provide processing rules for both senders and receivers of SEND messages employing the new options, as well as example negotiation message flows.

---

3. Supported Signature Algorithm Option

TOC

The Supported Signature Algorithm NDP option contains a list of signing and hashing algorithm pairs that the sender node supports. The format of this option is described in Figure 1 (Supported Signature Algorithm option):

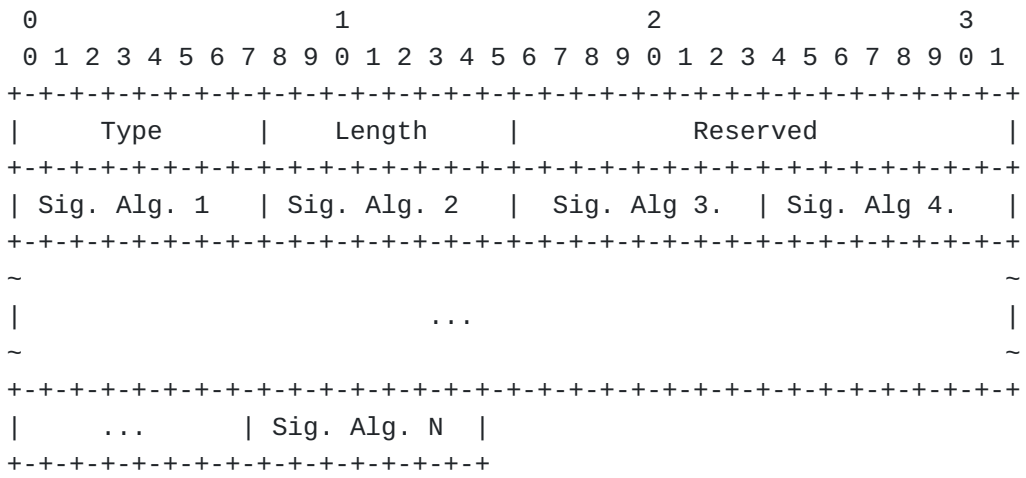


Figure 1: Supported Signature Algorithm option

Type
NDP option type, TBA. See Section 8 (IANA Considerations).

Length
The length of the option (including the Type, Length fields), in octets. 8-bit unsigned integer, the values lower that 2 are invalid.

Reserved
Reserved for future use. This 16-bit field MUST be set to zero by the sender, and MUST be ignored by the receiver.

Signature Algorithm

A one-octet long field indicating a signature algorithm and the corresponding hash algorithm that this node supports; this support implies at least ability to verify signatures of this PK algorithm.

The first leftmost bit, bit 0, if set to 0, indicates that the emitter is able to perform signature checks only (i.e. no signature generation with this type of signature algorithm). If this bit is set to 1, it indicates that the emitter has a public key of this type and can generate signatures. Bit 1 and 2 are reserved. Bit 3 to 7 are named Signature Type Identifier subfield and encode an identifier for the signature algorithm and corresponding hash algorithm. Default values for the Signature Type Identifier subfield defined in this document are taken in part from the IANA-defined numbers for the IKEv2 protocol, i.e. IANA registry named "IKEv2 Authentication Method":

- *Value 0 is RSA/SHA-1
- *Value 1 is RSA/SHA-256
- *Value 9 is ECDSA with SHA-256 on the P-256 curve
- *Value 10 is ECDSA with SHA-384 on the P-384 curve
- *Value 11 is ECDSA with SHA-512 on the P-521 curve

The Signature/hash Algorithm combinations SHOULD be included in order of preference.

A SSA option MAY be built to respect a Local Policy. However, the SSA option MUST not indicate Signature Algorithm(s) that the emitting node's CGA does not support and MUST contain at least one Signature Algorithm with the first bit on (i.e. this Signature Algorithm is available for signature generation).

3.1. Neighbor Cache interactions

TOC

Neighbor Cache MUST have the ability to store Supported Signature Algorithm information for each entry (i.e. IPv6 address). Supported Signature Algorithm information for an entry MAY be empty (e.g. entry created by a RFC 3971 node or an unverifiable message).

3.2. Processing Rules for Senders

TOC

If a node has been configured to use SEND, then all Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisement, and Redirect messages it sends MUST contain the Supported Signature Algorithm option. This option MUST contain in the Signing Algorithm field all the signature algorithms it is willing to use in signature generation and verification.

3.3. Processing Rules for Receivers

TOC

Upon receiving a SEND packet with a Supported Signature Algorithm Option, a receiver performs the following operations:

* when the message is a Neighbor Solicitation or a Router Solicitation, the receiving node computes the intersection between the set of Supported Signature Algorithm indicated by the option and its own. If the set is empty, this means the node will not be able to use a Signature Algorithm that the initiating node can check. Given the local policy, a receiver node MAY still respond to the received message using its "preferred" Signature Algorithm (even if the node knows the receiver will not be able to verify the Signature Algorithm). If the set is not empty, the receiving node will choose among the set one of the algorithms in order to generate a Universal Signature Option.

* If the message pass the SEND verifications (CGA verification, Timestamp, Nonce, Universal Signature Option verification) and contains a Supported Signature Algorithm Option, the information of the Supported Signature Algorithm in the Neighbor Cache is updated by the information contained in the Supported Signature Option attached to the message.

* If the message does not pass the SEND verifications because of a unverifiable RSA Signature Option or Universal Signature Option, if it contains a Supported Signature Algorithm Option, and the Neighbor Cache entry associated to that node does not contain any information about the Supported Signature Algorithm, the Neighbor Cache entry SHOULD be updated with the information contained in the Supported Signature Algorithm Option.

TOC

4. SEND Universal Signature Option

We propose replacing the RSA Signature Option by a new algorithm-independent signature option. The "Universal Signature Option" is an updated version of the RSA Signature Option, that allows a node to specify which of its potential multiple keys it is using. To achieve this, we use the 16-bit reserved field of the RSA Signature Option, and define a new 8-bit field that contains the position of the Public Key associated with the signature and a new 5-bit Signature Type Identifier field that details the type of algorithms used to generate the Digital Signature.

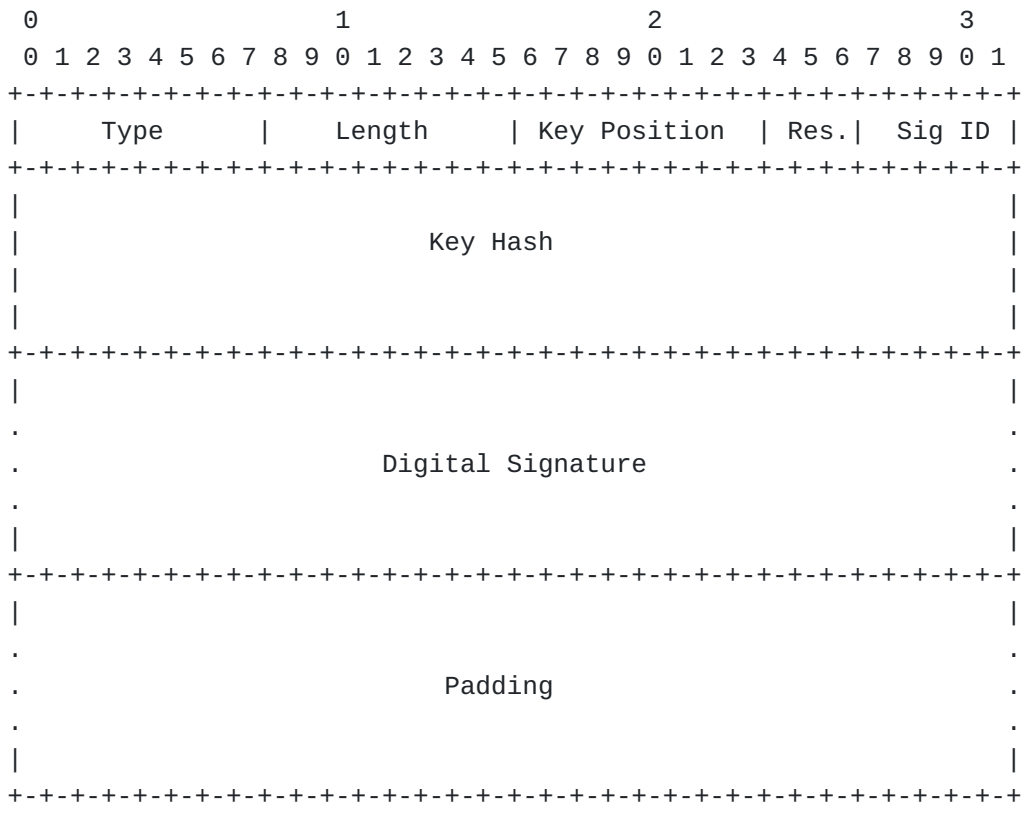


Figure 2: Universal Signature Option format

Type

Same value as in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.): 12.

Length

The length of the option (including the Type, Length,

Reserved, Key Hash, Digital Signature, and Padding fields) in units of 8 octets.

Key Position

An 8-bit field indicating which Public Key in the CGA parameter structure (carried in the CGA option) has been used to compute the Digital Signature. The index starts at 0, meaning the key is the one in the Public Key field. Values greater than 1 refer to Public Key found in the CGA Extension field (as defined in the companion document [cheneau-cga-pk-agility] (Cheneau, T., Laurent-Maknavicius, M., Shen, S., and M. Vanderveen, "Support for Multiple Signature Algorithms in Cryptographically Generated Addresses (CGAs)," June 2009.)]). Value 255 is a reserved value that indicates no CGA option in the message contains the Public Key.

Reserved

A 3-bit field reserved for future use. The value MUST be set to zero by the sender and MUST be ignored by the receiver.

Signature Type Identifier

Signature Type Identifier is a 5-bit field. It corresponds to the Signature Type Identifier subfield (bits 3 to 7 of the Signature Algorithm field) in the Supported Signature Algorithm option . It indicates the type of signature contained in the Digital Signature field.

Key Hash

A 128-bit field containing the most significant (leftmost) 128 bits of a hash of the public key used for constructing the signature. It is computed using the same hash function as used in generating digital signature (indicated in Signature Type Identifier). The hash value is computed over the presentation used in the Public Key field of the CGA Parameters data structure carried in the CGA option. Its purpose is to associate the signature with a particular key known by the receiver. Such a key can either be stored in the certificate cache of the receiver or be received in the CGA option in the same message.

Digital Signature

A variable-length field containing a signature constructed by using the sender's private key associated to the public key pointed by the Key Position field. The signature type is determined from the value of the Signature Type Identifier field. If the value of the Signature Type Identifier field is 0, then the Key Position field must be set to 0 and this Digital Signature field is computed the same way as the Digital Signature field of the RSA Signature Option described in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.). If the value of the Signature

Type Identifier field is 1, then this Digital Signature field is computed the same way as the Digital Signature field of the RSA Signature Option described in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.) except that the signature is computed with the RSASSA-PKCS1-v1_5 algorithm as defined in [PKCS1] (RSA Laboratories, "RSA Encryption Standard, Version 2.1," November 2002.) and hash function is SHA-256. If the value of the Signature Type Identifier field is 9, 10 or 11, then this Digital Signature field is computed using the ECDSA signature algorithm (as defined on [SEC1] (Standards for Efficient Cryptography Group, "SEC 1: Elliptic Curve Cryptography," September 2000.)) and hash function defined in Signature Type Identifier on the following data:

1. The 128-bit CGA Message Type tag [RFC3972] (Aura, T., "Cryptographically Generated Addresses (CGA)," March 2005.) value for SEND, 0x086F CA5E 10B2 00C9 9C8C E001 6427 7C08. (The tag value has been generated randomly by the editor of the [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.) specification.).

2. The 128-bit Source Address field from the IP header.

3. The 128-bit Destination Address field from the IP header.

4. The 8-bit Type, 8-bit Code, and 16-bit Checksum fields from the ICMP header.

5. The NDP message header, starting from the octet after the ICMP Checksum field and continuing up to but not including NDP options.

6. All NDP options preceding, but not including, any of the Universal Signature options.

This field starts after the Key Hash field. The length of the Digital Signature field is determined by the length of the Universal Signature option minus the length of the other fields (including the variable length Pad field).

Padding This variable-length field contains padding, as many bytes long as remain after the end of the signature.

A Neighbor Solicitation/Advertisement, Router Solicitation/ Advertisement and Redirect message MAY contain more than one Universal Signature Option, as long as it does not exceed the MTU. This is particularly useful for routers operating in heterogeneous networks, where hosts have a disjoint set of supported signature algorithms. For

information on how to compute the message size, see Appendix A (On the number of Public Keys supported per CGA).

4.1. Processing Rules for Senders

TOC

When sending a SEND message spontaneously, an emitter node CAN choose a signature algorithm of its preference (defined by its local policy) among the corresponding Public Keys carried in the CGA option. Using this signature algorithm, the node computes the Digital Signature and fills the Key Position field with the position of the key in the CGA parameter data structure.
If the node has been configured to use SEND, then all Neighbor Solicitation, Neighbor Advertisement, Router Advertisement, and Redirect messages MUST contain at least one Universal Signature option. Router Solicitation messages not sent with the unspecified source address MUST contain the Universal Signature option.
A node sending a message with one or more Universal Signature option(s) MUST construct the message as follows:

- If the node has previously received hints (e.g. a NDP message with a Supported Signature Algorithm option or an entry in the Neighbor Cache) on the type of Signature Algorithm it should use, it MUST restrict its choice on those Signature Algorithms.

- The message is then constructed in its entirety, without any of the Universal Signature options.

- The Universal Signature option(s) is (are) added as the last option in the message.

- The data to be signed is constructed as explained in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.), under the description of the Digital Signature field.

- The message, in the form defined above, is signed by using the configured private key associated to the selected Signature Algorithm, and the result signature is is encapsulated into the Digital Signature field.

TOC

4.2. Processing Rules for Receivers

Neighbor Solicitation, Neighbor Advertisement, Router Advertisement, and Redirect messages without any Universal Signature option or with an unverifiable Universal Signature option MUST be treated as unsecured (i.e., processed in the same way as NDP messages sent by a non-SEND node). See Section 8 of [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.).
Router Solicitation messages without any Universal Signature option MUST also be treated as unsecured, unless the source address of the message is the unspecified address.
Redirect, Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, and Router Advertisement messages containing one or more Universal Signature option MUST be checked as follows:

- The receiver MUST ignore any options that come after the first Universal Signature option. (The options are ignored for both signature verification and NDP processing purposes.)

- The Key Hash field MUST correspond to a known public key, either one learned from the CGA option in the same message by the position indicated in the Key Position field message, or one known by other means.

- The Digital Signature field MUST have correct encoding and MUST not exceed the length of the Universal Signature option minus the Padding.

- The Digital Signature verification MUST show that the signature has been calculated as specified in the previous section.

- If the use of a trust anchor has been configured, a valid certification path (see Section 6.3 of [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.)) between the receiver's trust anchor and the sender's public key MUST be known.

Messages that do not pass all the above tests MUST be silently discarded if the host has been configured to accept only secured ND messages. The messages MAY be accepted if the host has been configured to accept both secured and unsecured messages but MUST be treated as unsecured messages. The receiver MAY also otherwise silently discard packets (e.g., as a response to an apparent CPU exhausting DoS attack).

---

5. Basic negotiation

TOC

5.1. Overview

TOC

This section describes different configuration of SEND-enabled nodes with varying signing capabilities and their interaction during the negotiation phase.
Case 1: when both nodes support the same two Signature Algorithms, they can pick the Signature Algorithm they prefer for signing and are able to verify each others signature. Figure 3 (Basic negotiation - Case 1) is an example of such a message flow.

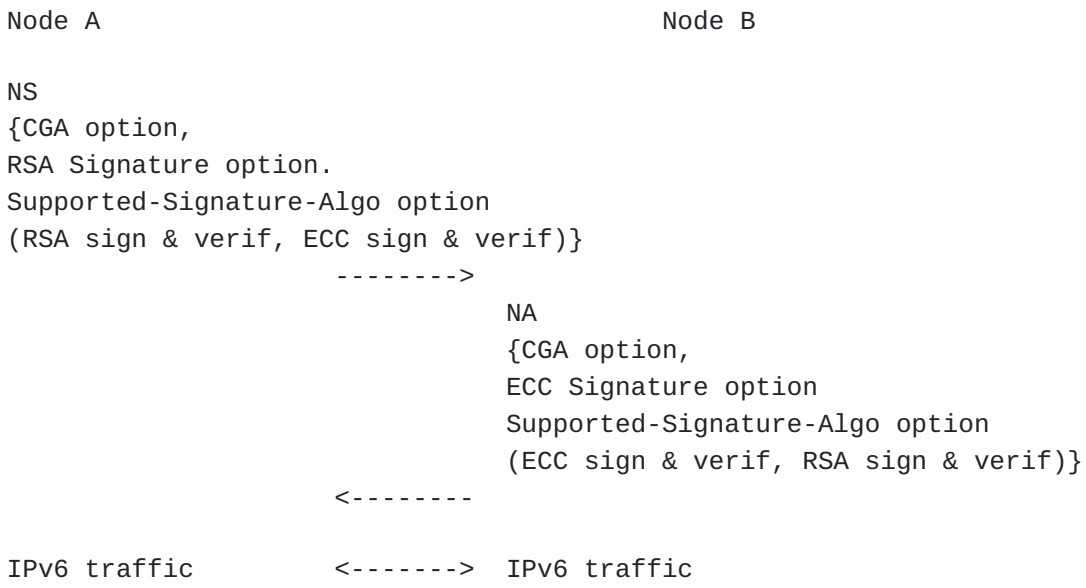


Figure 3: Basic negotiation - Case 1

Case 2: two nodes sharing at least one common Signing Algorithm must be able to securely communicate. Figure 4 (Basic negotiation - Case 2) is an example of such a message flow.

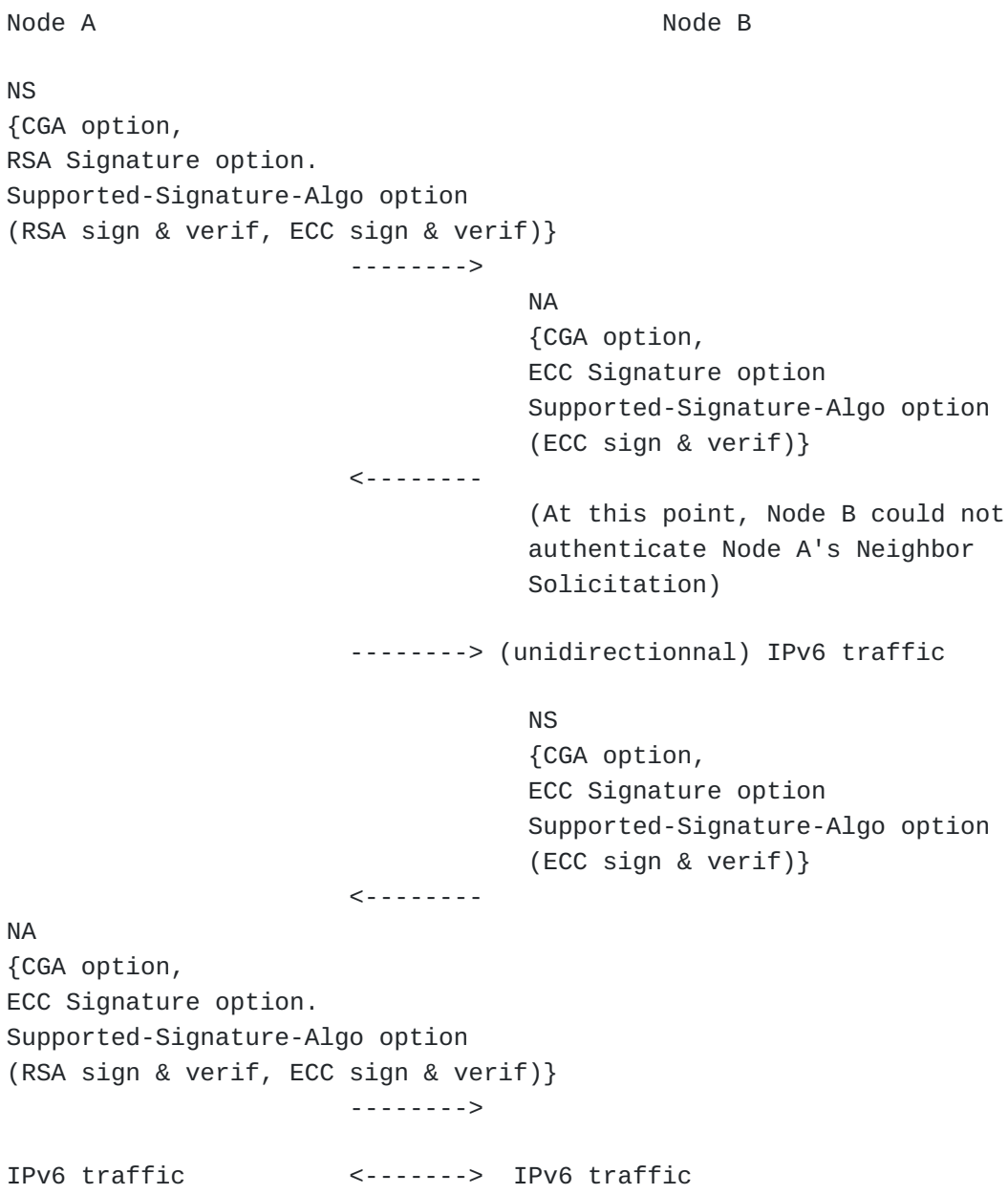


Figure 4: Basic negotiation - Case 2

Case 3: when two nodes have a disjoint set of Signature Algorithm support for signing, but the two nodes are able to verify each others, a full negotiation is possible. Figure 5 (Basic negotiation - Case 3) is an example of such a message flow.

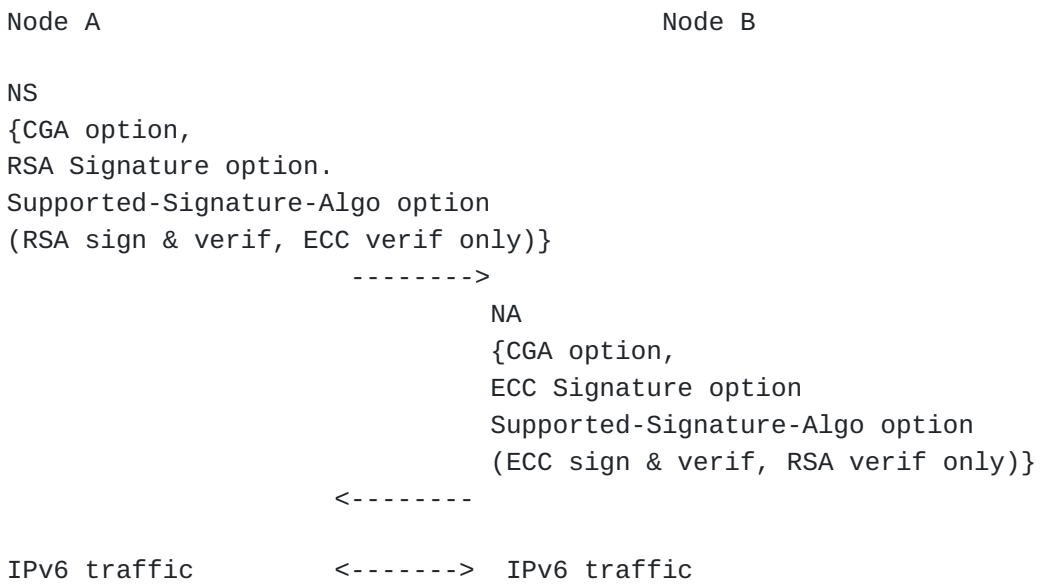


Figure 5: Basic negotiation - Case 3

Case 4: when two nodes have a disjoint set of Signature Algorithm support for signing, but one node is able to verify, a partial negotiation is possible. Figure 6 (Basic negotiation - Case 4) is an example of such a message flow.

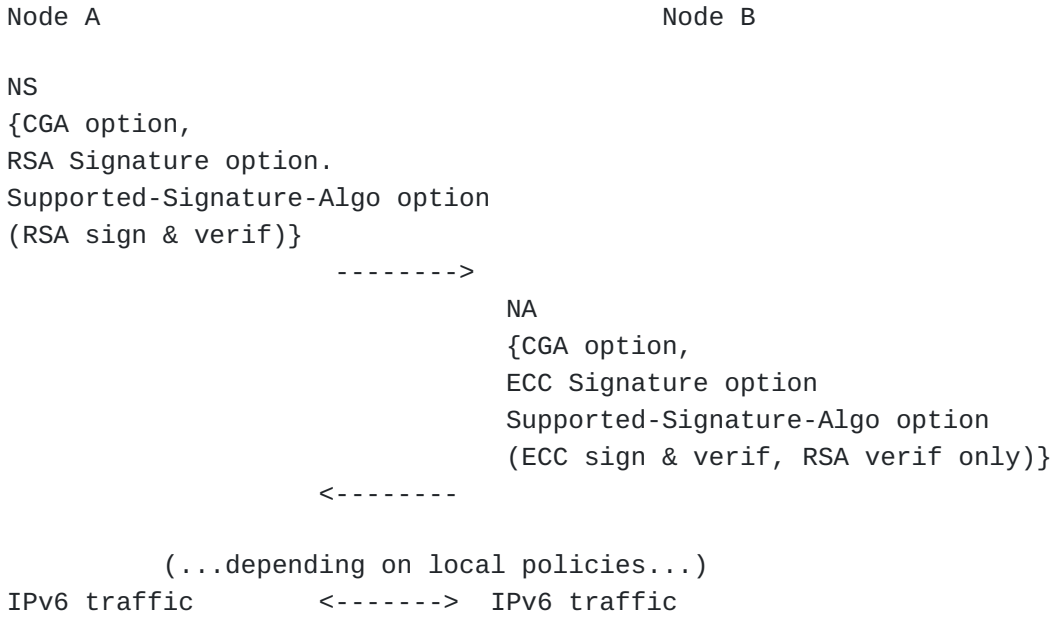


Figure 6: Basic negotiation - Case 4

Section 6 (Router-as-a-notary function) describes an optional functionality that allow nodes in Case 4 to perform a trustful complete negotiation.

5.2. Sending Unsolicited Messages

TOC

When sending unsolicited message, a node MAY have to rely on the entries of its Neighbor Cache. The Neighbor Cache will provide hints concerning the Signature Algorithm supported by the neighbors. Neighbor Cache can assist the node in the Signature Algorithm selection process when:

*A router advertises unsolicited Router Advertisement message to the All-Nodes multicast address (e.g. to indicate a prefix lifetime is going down to 0). The router needs to know which signature algorithm(s) to use in order to send verifiable messages to hosts. To do so, the router MAY rely on the Neighbor Cache and compute an intersection of the set of all Supported Signature Algorithms. The router will then be able to advertise a Router Advertisement signed multiple times with the resulting subset of Supported Signature Algorithms or advertise multiple Router Advertisements, each signed with a single Signature Algorithm part of the intersection.

*A node sends unsolicited Neighbor Advertisement (e.g. when changing its Link-Layer address). This is similar to the previous problem and can also be solved using the Neighbor Cache the same way.

*A router sends a Redirect message to a host. Choosing a supported signature algorithm without probing the node can be difficult. However, Neighbor Cache will most likely contain an entry for the host, prior to the decision to send a Redirect message, because of the Address Resolution process. This entry should contain information on the Supported Signature Algorithm(s) and thus provide hints concerning the Signature Algorithm to choose to sign the Redirect messages.

Note that the information on the neighbors with which a communication has occurred recently or is ongoing are in the Neighbor Cache and are maintained up to date through the Neighbor Unreachability Detection procedure.

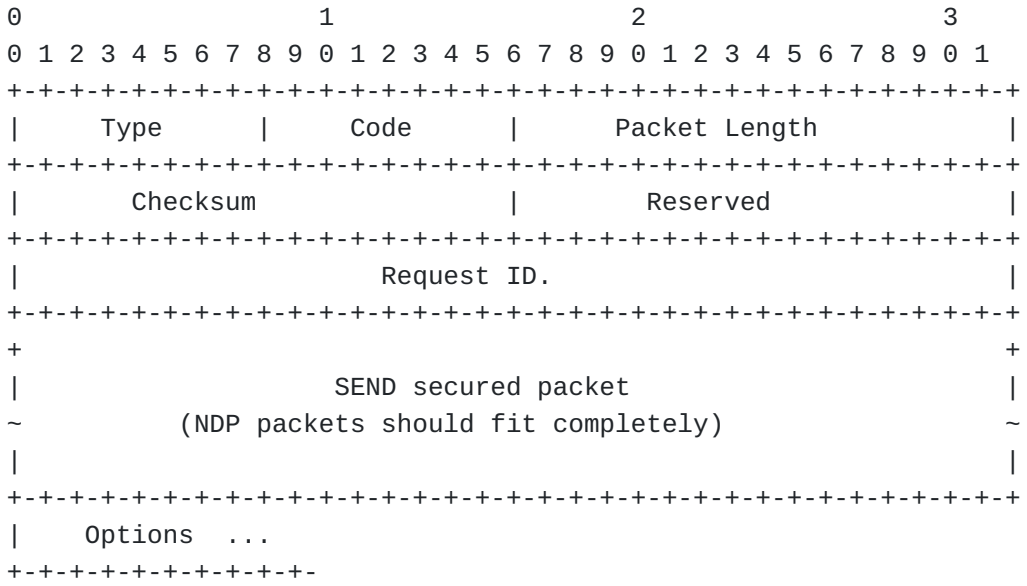
TOC

6. Router-as-a-notary function

This optional functionality enhances backward compatibility by introducing a new entity. This new entity, named "notary", certifies the authenticity of a node's message. This improves communication when, for example, two nodes have a disjoint set of supported Signature Algorithm types and still require secure neighbor discovery.
In this specification, the notary function is offered by routers, although other nodes may offer this capability in the future specification. Authorization for the router to act as a notary is shown through router's certificate in a KeyPurposeID as defined in [krishnan-cgaext-send-cert-eku] (Krishnan, S., Kukec, A., and K. Ahmed, “Certificate profile and certificate management for SEND,” March 2009.) and provided by the trust anchor.
The notary function requires the two specific ICMP messages: signature check request message and signature status message.

6.1. Signature Check Request Message

TOC



Signature Check Request Message format

Type
TBA.

Code

TBA.

Packet Length

Packet length is the size of the SEND secured packet

Checksum

Checksum is a CRC-16 of the whole packet. During the CRC-16 computation, this field is set to 0. The purpose of this field is to quickly invalidate transmission errors.

Reserved

This 16-bit field is reserved. MUST be set to 0 by senders and ignored by receivers.

Request Identifier

Request Identifier helps matching a signature check request and the signature status (response) messages. Request Identifier field is randomly generated.

SEND secured packet

SEND secured packet is the packet that the node was not able to verify on his own, subject of the verification. Note that the encapsulated packet MUST not make the whole Signature Check Request message exceed the MTU (as no fragmentation support is available).

Options

This field contains one or more NDP options. Currently, only one option is mandatory in this field. It is the Supported Signature Algorithm option, that allows the notary to choose a correct signature algorithm to sign the Signature Status message.

Note that this message MAY be protected by usual SEND NDP options (CGA option, Timestamp, Nonce, Universal Signature Option). In this case, the Universal Signature Option contains the whole packet that the node wants to be checked on the router (so packet may not be tampered with). A router acting as a notary processes the packet as follows:

if the packet is protected with SEND options, the notary:

*Verifies the CGA of the emitter

*Verifies the Universal Signature Option of the message (linked to CGA of the source address). If more than one Universal Signature Options are in the message, the notary can decide to check any of them.

*Verifies the CGA and signature of the SEND secured packet (inner packet).

*Responds with a Signature status message (defined in the following section) indicating the status of the SEND secured packet Universal Signature Option.

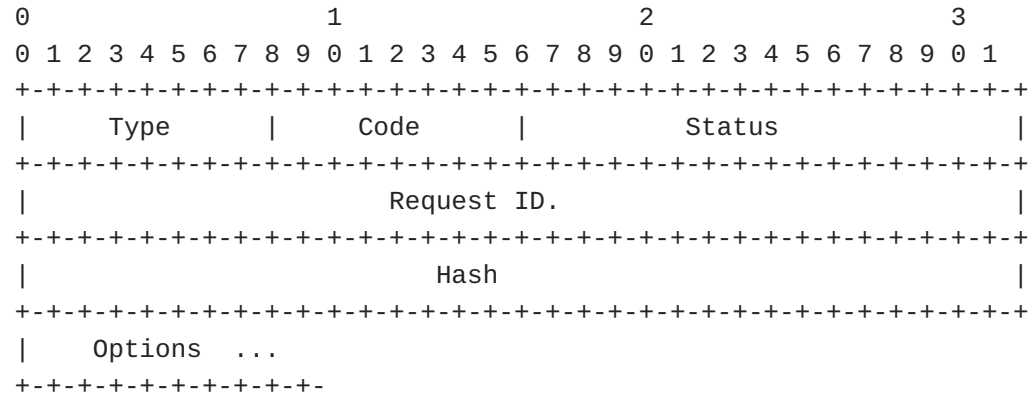
if the packet is not protected, the notary:

*verifies the CGA and signature of the SEND secured packet (inner packet). If more than one Universal Signature Option are in the message, the notary can decide to check any of them.

*Responds with a Signature status message (defined in the following section) indicating the status of the SEND secured packet Universal Signature Option.

6.2. Signature Status Message

TOC



Signature Status Message format

Type

TBA.

Code

TBA.

Status

The 16-bit status field can be set to any of the following values:

> 0: all validation checks passed
>
> 1: Signature Check Request message checksum failed
>
> 2: inner packet CGA verification check failed
>
> 3: inner packet signature verification check failed
>
> 4: unsupported hash algorithm (to compute Hash1/Hash2)
>
> 5: unsupported Public Key algorithm
>
> 6: ask later (router is busy)

Request Identifier

The Request Identifier helps matching a signature check request and the signature status (response) message. The Request Identifier is copied from the Signature Check Request message.

Hash

The Hash field contains the result of a hash function applied on the Request ID field and on the Send Secured Packet field of the Signature Check Request message. The hash function is the same as the one in the Key Hash field of the Universal Signature Option that will protect this message.

Options

This field contains one or more NDP options. Mandatory options are CGA Option, Timestamp Option and Universal Signature Option. Universal Signature Option MUST be the last option.

This message is a response to a Signature Check Request message and is protected by SEND options generated using a public key contained in a certificate of the router authorized to act as notary. If the Signature Check Request message is protected by the Nonce option, this option MUST be copied in the Signature Status message.
On reception of this message, a requesting node performs CGA verification, Nonce (if included in the initial request) and Timestamp checks, and Universal Signature Option check. If any of those test

fails, the packet is dropped and an error MAY be logged. Then, if the status message is 0, that node can treat the original packet that created the need for a Notary Signature Check Request message as a secured packet. On a status value different from 0, the packet will be considered as unsecure and be treated as such. Status value MAY be loged for further diagnosis.

6.3. Using notary for DAD procedure

TOC

When performing the DAD procedure, a node can receive Neighbor Solicitation or Neighbor Advertisement that are protected by a Universal Signature Option the node can not check. In this specific case, the node can ask the notary to check the signature for him. However, the node, while performing DAD, MUST send the Signature Check Request message using the unspecified address as source address. The notary MUST respond with a Signature Status message directed to the All-Node multicast address.

---

7. Security Considerations

TOC

RSA key length (bits)	ECC key length (bits)
3072	256
7680	384
15380	512

Equivalence between Elliptic Curves and RSA security levels

Table 1: Security level equivalence between ECC and RSA

Section 4 (SEND Universal Signature Option) presents a new Universal Signature Option. A recommended use of this option is to allow signatures of equivalent security level (i.e. Public Keys with equivalent key lengths) as shown in Table 1 (Security level equivalence between ECC and RSA). See also section 4 of the companion draft [cheneau-cga-pk-agility] (Cheneau, T., Laurent-Maknavicius, M., Shen, S., and M. Vanderveen, “Support for Multiple Signature Algorithms in Cryptographically Generated Addresses (CGAs),” June 2009.).

Usage of SHA-1 for signature is strongly NOT RECOMMENDED, and when available should be preferred by the usage of SHA-256. SHA-1 security is been proved to be flawed in the light of recent attacks [Recent SHA-1 Attack] (McDonald, C., Haukes, P., and J. Pieprzyk, “SHA-1 collisions now 2^52,” May 2009.) [NIST-st] (National Institute of Standards and Technology, “NIST Comments on Cryptanalytic Attacks on SHA-1,” .).

The Universal Signature Option is vulnerable to downgrade attacks. That is, given that a node can employ multiple signature types, an attacker may choose to use a flawed one. To mitigate this issue, nodes are allowed, on a local policy, to refuse to check certain types of signature (i.e. those which are know to be flawed) and will treat the associated messages as unsecured. When trying to completely mitigate downgrade attacks, an administrator MAY deploy SEND-secured nodes only authorizing a single signature algorithm scheme. This comes at a price of a reduced interoperability.

Section 6 (Router-as-a-notary function) introduces an optional notary functionality that offers to nodes to check messages on their behalf, involving heavy cryptographic computation. This can lead to flooding attacks and Denial of Services. However, Neighbor Discovery Protocol [RFC4861] (Narten, T., Nordmark, E., Simpson, W., and H. Soliman, “Neighbor Discovery for IP version 6 (IPv6),” September 2007.) and Secure Neighbor Discovery Protocol [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.) are already prone to flooding attacks. One possible solution is to use rate limiting on Signature Check Request messages.

Notary functionality is also vulnerable to "Good Router Goes Bad" attacks (as described in [RFC3756] (Nikander, P., Kempf, J., and E. Nordmark, “IPv6 Neighbor Discovery (ND) Trust Models and Threats,” May 2004.)). Notary can make node trust unsecured packets and drop valid ones. This issue can be mitigated when multiple notaries are present on a link. The node can use a round-robin algorithm to load-balance the Signature Check Request message, thus reducing the risk of cache poisoning by a compromised notary.

---

8. IANA Considerations

TOC

This document requests IANA to allocate types for the two new notary ICMP messages.

Section 3 (Supported Signature Algorithm Option) defines a Signature Type Identifier subfield containing new values corresponding to different Signature Algorithm. This document requests creation of a new registry to the IANA.

TOC

9. Acknowledgments

The authors gratefully acknowledge the contributions of Marcelo Bagnulo, Gabriel Montenegro, Greg Daley, Dave Thaler, Steve Kent, Jari Arko, and Francis Dupont for their helpful feedback.

---

10. References

TOC

10.1. Normative References

TOC

[RFC3972]	Aura, T., “Cryptographically Generated Addresses (CGA),” RFC 3972, March 2005 (TXT).
[RFC3971]	Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” RFC 3971, March 2005 (TXT).
[RFC4982]	Bagnulo, M. and J. Arkko, “Support for Multiple Hash Algorithms in Cryptographically Generated Addresses (CGAs),” RFC 4982, July 2007 (TXT).
[cheneau-cga-pk-agility]	Cheneau, T., Laurent-Maknavicius, M., Shen, S., and M. Vanderveen, “Support for Multiple Signature Algorithms in Cryptographically Generated Addresses (CGAs),” draft-cheneau-cga-pk-agility-01 (work in progress), June 2009 (TXT).

10.2. Informative References

TOC

[RFC2460]	Deering, S. and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification,” RFC 2460, December 1998 (TXT, HTML, XML).
[RFC3756]	Nikander, P., Kempf, J., and E. Nordmark, “IPv6 Neighbor Discovery (ND) Trust Models and Threats,” RFC 3756, May 2004 (TXT).
[RFC4581]	Bagnulo, M. and J. Arkko, “Cryptographically Generated Addresses (CGA) Extension Field Format,” RFC 4581, October 2006 (TXT).
[RFC4861]	Narten, T., Nordmark, E., Simpson, W., and H. Soliman, “Neighbor Discovery for IP version 6 (IPv6),” RFC 4861, September 2007 (TXT).
[NIST-st]	National Institute of Standards and Technology, “NIST Comments on Cryptanalytic Attacks on SHA-1.”

[krishnan-cgaext-send-cert-eku]	Krishnan, S., Kukec, A., and K. Ahmed, “Certificate profile and certificate management for SEND,” draft-krishnan-cgaext-send-cert-eku-03 (work in progress), March 2009 (TXT).
[FIPS-186-3]	National Institute of Standards and Technology, “Draft Digital Signature Standard,” FIPS PUB 186-3, March 2006.
[PKCS1]	RSA Laboratories, “RSA Encryption Standard, Version 2.1,” PKCS 1, November 2002.
[FIPS.180-2]	National Institute of Standards and Technology, “Secure Hash Standard,” FIPS PUB 180-2, August 2002.
[SEC1]	Standards for Efficient Cryptography Group, “SEC 1: Elliptic Curve Cryptography,” September 2000.
[Recent SHA-1 Attack]	McDonald, C., Haukes, P., and J. Pieprzyk, “SHA-1 collisions now 2^52,” May 2009.

Appendix A. On the number of Public Keys supported per CGA

TOC

RSA key length (bits)	Public exponent	Size of the DER-encoded Public Key (bytes)
384	3 or 17	76
384	65537	78
512	3 or 17	92
512	65537	94
1024	3 or 17	160
1024	65537	162
2048	3 or 17	292
2048	65537	294
3072	3 or 17	420
3072	65537	422
7680	3 or 17	996
7680	65537	998
15360	3 or 17	1956
15360	65537	1958

Table 2: Common sizes for DER-encoded RSA Public Key

RSA Key Length (in bits)	Size of the Digital Signature field without padding
384	48
512	64
1024	128
2048	256
3072	384
7680	960
15360	1920

Table 3: Common sizes of the Digital Signature field when using RSA

Name of the elliptic curve	Size of the DER-encoded Public Key (bytes)
P-256	88
P-384	120
P-521	158

Table 4: Common sizes for DER-encoded ECC Public Key

Name of the elliptic curve	Size of the Digital Signature field (without padding)
P-256	71
P-384	104
P-521	139

Table 5: Common sizes of the Digital Signature field when using ECDSA (+ DER encoding)

When using multiple public keys to form a CGA, one may reach the maximum number of possible public keys before each Neighbor Discovery Message exceed the Maximum Transfer Unit (which must be at least 1280 octets according to [RFC2460] (Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," December 1998.)). This section aims to approximate this limit.
Numerous factors (presence and number of option, size of public keys, etc) influence the size of the Neighbor Discovery message. For example, when sending a SEND-secured Router Advertisement message:

*The IPv6 header is 40 bytes long. Described in [RFC2460] (Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," December 1998.).

*The bare Router Advertisement message (without any option) is 16 bytes long. Described in [RFC4861] (Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," September 2007.).

*A Prefix Information Option (can appear more than once) is 32 bytes long. Described in [RFC4861] (Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," September 2007.).

*A Source Link-Layer Option, when a IEEE 802 address is used, is 8 bytes long. Described in [RFC4861] (Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," September 2007.).

*A MTU Option is 8 bytes long. Described in [RFC4861] (Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," September 2007.).

*The CGA Option is the size of the CGA Parameter Data Structure plus 4 bytes rounded up to the closest multiple of 8 value. This option is defined in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.). The CGA Parameter Data Structure (defined in [RFC3972] (Aura, T., "Cryptographically Generated Addresses (CGA)," March 2005.) size depends on the following fields:

  -Modifier: 16 bytes long.

  -Subnet Prefix: 8 bytes long.

  -Collision Count: 1 byte long.

  -Public Key: variable size. Table 2 (Common sizes for DER-encoded RSA Public Key) provides size of the commonly used DER-encoded RSA Public Keys. Table 4 (Common sizes for DER-

encoded ECC Public Key) provides size for the commonly used DER-encoded ECC Public Keys.

-Extension(s): variable size. Public Key Extension field defined in [cheneau-cga-pk-agility] (Cheneau, T., Laurent-Maknavicius, M., Shen, S., and M. Vanderveen, “Support for Multiple Signature Algorithms in Cryptographically Generated Addresses (CGAs),” June 2009.) is 4 bytes plus the size of the Public Key long. Public Key size are defined in Table 2 (Common sizes for DER-encoded RSA Public Key) and Table 4 (Common sizes for DER-encoded ECC Public Key).

*The Timestamp Option is 16 bytes long. Defined in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.).

*The Nonce Option minimum size is 8 bytes long. Defined in [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.).

*The Universal Signature Option depends on the size of the Digital Signature. The fixed part of the option is 20 bytes long. This option is updated in this document. Table 3 (Common sizes of the Digital Signature field when using RSA) presents common sizes for usual Digital Signature field when using RSA. Table 5 (Common sizes of the Digital Signature field when using ECDSA (+ DER encoding)) presents common sizes for Digital Signature field when using ECDSA. This option size must be a multiple of 8 bytes.

A Router Advertisement message, carrying a Prefix Information Option and a Source Link-Layer Option, without Nonce, with one 1024-bits long RSA Public Key and a Public Exponent of 3 in the CGA Option is 456 bytes long. Using the same RSA Public Key, adding one ECC P-521 key to CGA Option, the same message, signed with a Universal Signature option generated by RSA and a Universal Signature Option signed by ECDSA, is 768 bytes long. Note that EC P-521 and 1024-bits RSA keys should not be used together because they do not present the same security level (see Section 7 (Security Considerations)) and are shown here to indicate sizes of messages with "big" keys.

Authors' Addresses

TOC

	Tony Cheneau
	Institut TELECOM, TELECOM SudParis, CNRS SAMOVAR UMR 5157
	9 rue Charles Fourier
	Evry 91011
	France

Email:	tony.cheneau@it-sudparis.eu
	Maryline Laurent-Maknavicius
	Institut TELECOM, TELECOM SudParis, CNRS SAMOVAR UMR 5157
	9 rue Charles Fourier
	Evry 91011
	France
Email:	maryline.maknavicius@it-sudparis.eu
	Sean Shen
	Huawei
	No. 9 Xinxi Road
	Beijing 100085
	China
Email:	sshen@huawei.com
	Michaela Vanderveen
	Qualcomm
Email:	mvandervn@gmail.com