CGA & SEND maintenance	T. Cheneau	
Internet-Draft	M. Maknavicius	
Updates: RFC3971	TMSP	
(if approved)	S. Shen	
Expires: August 25, 2009	Huawei	
	M. Vanderveen	
	Qualcomm	
	February 21, 2009	

TOC

Signature Algorithm Agility in the Secure Neighbor Discovery (SEND) Protocol
draft-cheneau-send-sig-agility-00

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

Cryptographically Generated Addresses (CGA) [RFC3972] (Aura, T., “Cryptographically Generated Addresses (CGA),” March 2005.) have been designed primarily for securing Neighbor Discovery [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, “SEcure Neighbor Discovery (SEND),” March 2005.). At the time when they were specified, CGAs allowed only one signing algorithm, namely RSA. While mandating a single public key signing algorithm does help with interoperability, it does not address the issue of computational efficiency. It is well known that the RSA signature generation and verification is computationally expensive.

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 algorithm agility. Besides accommodating power-constrained nodes, signing 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 agility for nodes configured to perform secure neighbor discovery operations when attaching to a new link. 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 when attaching to a new link. 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),” Feb 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 and 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 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 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 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 router appear as having multiple identities.

Type R4 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.

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 and R2.

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 must 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 or R4) 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 two signature algorithms in common (one of which is likely preferred over the other), will be able to perform secure neighbor discovery with any of the two algorithms.

TOC

2.2. Agility Requirements

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 should be able to communicate using SEND and sign messages using the common Signature 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),” Feb 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 chose an appropriate signature algorithm. This solution encompasses the following:

*A "Supported Signature Algorithm" NDP option which contains a list of signing algorithms that the sender node supports for SEND purposes;

*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 algorithms that the sender nodes supports. The format of this option is described in Figure 1 (Supported Signature Algorithm option):

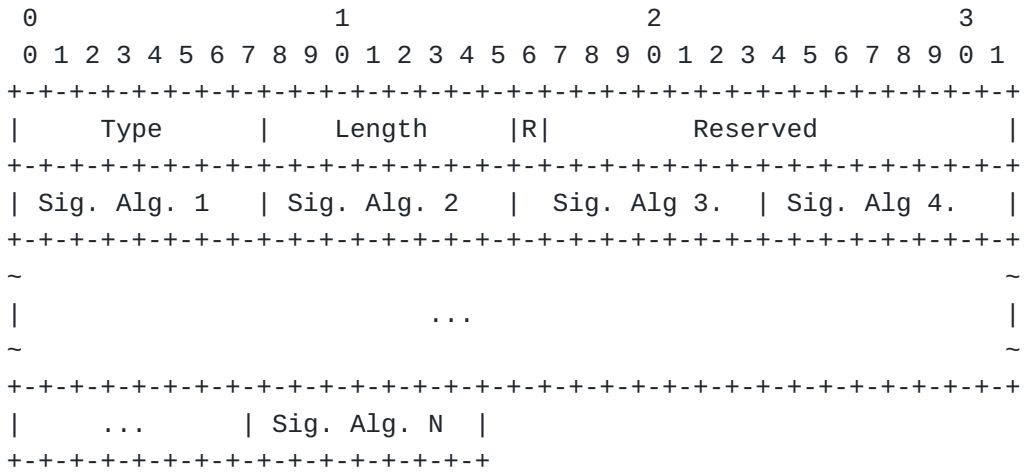


Figure 1: Supported Signature Algorithm option

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 value 0 is invalid.

R
"Resend" flag. If this bit is set, it indicates that the sender of this packet was not able to validate the packet that this packet was sent in response to. Spontaneous packets (i.e. those

not sent in response to a [request] packet) MUST leave this bit cleared.

Reserved

Reserved for future use. This 15-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 that is supported by the node, 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 on 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 the signature algorithm identifier. This signature algorithm identifier binds a Public Key algorithm with an hash algorithm. Default values for the Signature Type Identifier subfield defined in this document are:

*Value 1 is RSA/SHA-256

*Value 2 is ECDSA/SHA-256

*Value 0 is reserved for future use.

The Signature Algorithms SHOULD be included in order of preference.

3.1. 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 signature algorithms it is willing to use in signature verification.

TOC

3.2. Processing Rules for Receivers

Upon receiving a SEND packet with a Supported Signature Algorithm Option, a receiver checks the 'R' flag:

*when the 'R' flag is not set and the message is a Neighbor Advertisement or Router Advertisement, a host need not parse this option any further. A router MAY choose not to parse this option.

*when the 'R' flag is not set and the message is a Neighbor Solicitation, the receiving node computes the intersection between the set of Supported Signature Algorithms 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 will 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.

*when the 'R' flag is set, the receiver checks if it supports any of sender's supported signature algorithms. If more than one signature algorithms is found to be mutually supported, the receiver MAY decide to use the sender's most preferred one according to the order of appearance in the aforementioned NDP option. In any case, if at least one mutually supported signature algorithm exists, the receiver uses one of these algorithms to generate a Universal Signature Option for protection of the resent packet. This resent packet contains the same information that the other node couldn't verify (except for the signature). If the 'R' flag is set, and if no matching signature algorithm is found, the receiver processes the packet as if the 'R' flag was not set.

---

4. SEND Universal Signature Option

TOC

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-bits 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.

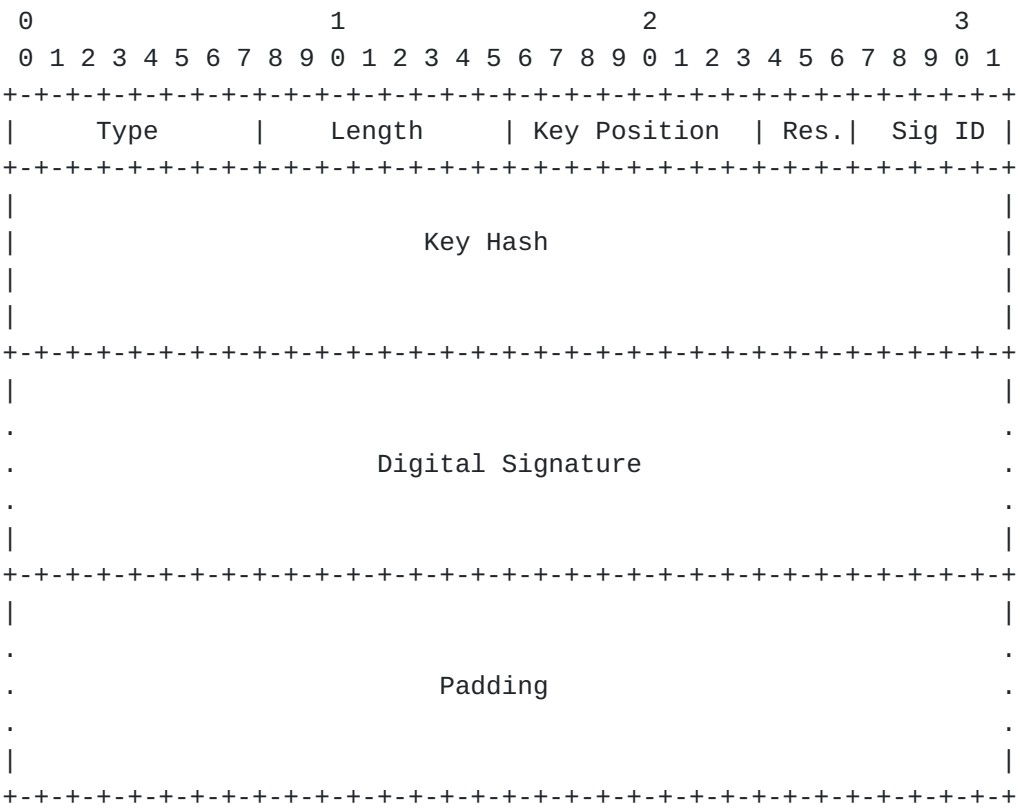


Figure 2: Signature Option format

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 over 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)," Feb 2009.)]).

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 Algorithm field in the Supported Signature Algorithm option. It indicates the type of Signature contained in the Digital Signature field.

Key Hash

The Key Hash field is a 128-bit field containing the most significant (leftmost) 128 bits of a hash function of the public key used. If the Signature Type Identifier value is 0 then this field is a is computed using SHA-1 value of the public key used for constructing the signature. This Key Hash is computed the same way as the Key Hash in RSA Signature Option described [RFC3971] (Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)," March 2005.). If the Signature Type Identifier value is different than 0 then this field is computed using SHA-256 [FIPS.180-2] (National Institute of Standards and Technology, "Secure Hash Standard," August 2002.) value of the public key used for constructing the signature. The SHA-256 hash 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 and the SHA-256 hash, as defined in

[PKCS1] (RSA Laboratories, “RSA Encryption Standard, Version 2.1,” November 2002.). If the value of the Signature Type Identifier field is 2, 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 SHA-256 on the following datas:

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.

TOC

4.1. Processing Rules for Senders

When sending a SEND message spontaneously or in response to message with the 'R' flag cleared in the Supported Signature Algorithm Option, an emitter node CAN choose a signature algorithm of its preference (defined by 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 MUST construct the message as follows:

*If the node as previously received hints (e.g. an NDP message with a Supported Signature Algorithm option and the 'R' flag on) on the type of Signature Algorithm it should use, it MUST restrain its choice on those Signature Algorithm. its choice on those Signature Algorithm.

*The message is 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 Figure 2 (Signature Option format), 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 resulting signature is put in the Digital Signature field. When using RSA, this signature is a PKCS#1 v1.5 signature. When using ECDSA, the signature value is as defined in [FIPS-186-3] (National Institute of Standards and Technology, “Draft Digital Signature Standard,” March 2006.). 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 Padding field).

TOC

4.2. Processing Rules for Receivers

Neighbor Solicitation, Neighbor Advertisement, Router Advertisement, and Redirect messages without any 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.

When checks fail due to an unsupported signature algorithm type, and if the Supported Signature Algorithm Option of the message shows that a common Signature Algorithm is available, the node MUST send back a packet to indicate to the emitter that the packet needs to be resent. Depending on the received packet, the node will have to send:

- A Router Solicitation if the message was a Router Advertisement or Redirect message; or

- A Neighbor Solicitation is the message was a Neighbor Advertisement or a Neighbor Solicitation (e.g. during the DAD procedure)

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).

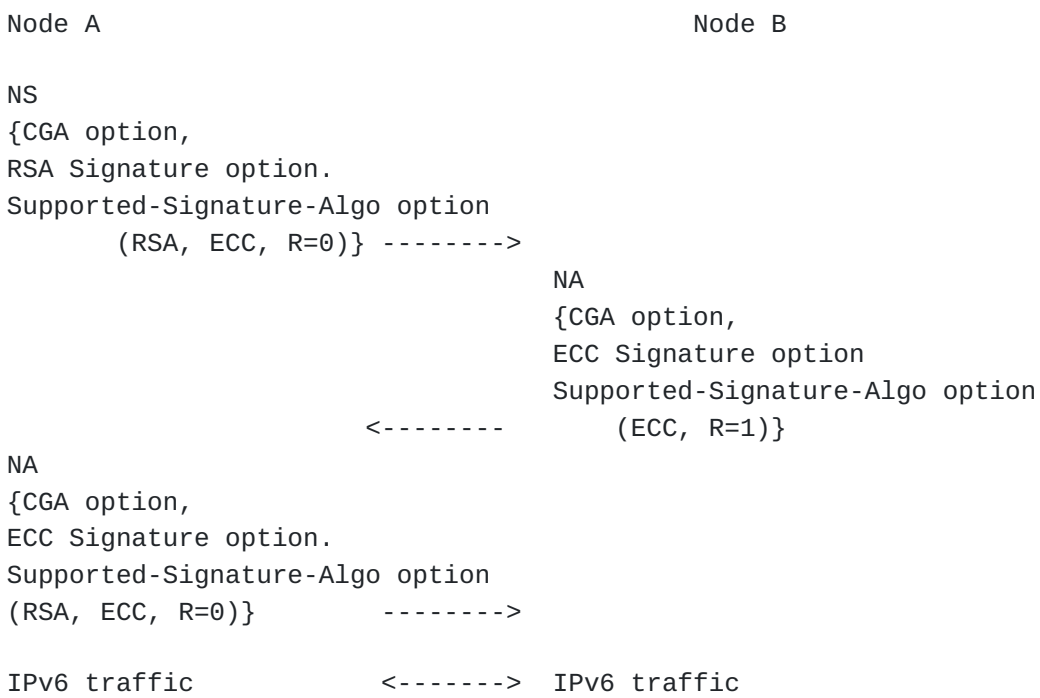
5. Basic negotiation

TOC

5.1. Overview

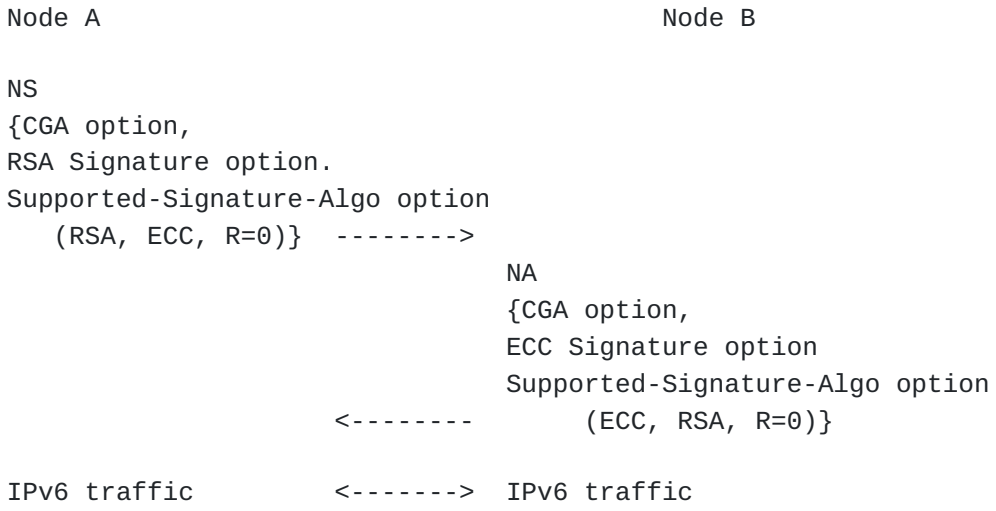
TOC

Two nodes sharing a common Signing Algorithm must be able to securely communicate. Below is an example of such a message flow.



Basic Negotiation- Case 1

When both nodes support the same two algorithms, then we have the following case:



Basic Negotiation- Case 2

6. Router-as-a-notary function

TOC

This optional functionality enhances backward compatibility by introducing a new entity. Here, the entity named "notary" serves to certify the authenticity of a node's message. This improves communication when 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.
Authorization for the router to act as a notary is provided through router's certificate (could be store 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,” November 2008.)) provided by the trust anchor.
The notary function requires the two specific messages: Signature check request and signature status.

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).

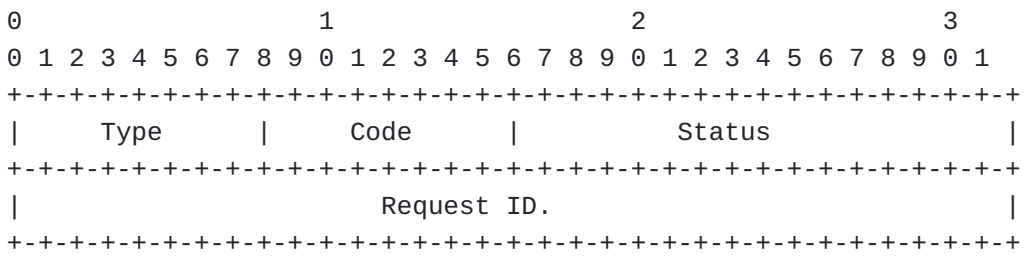
This message is protected by usual SEND NDP options (TS, Nonce, Signature). It contains the whole packet that the node wants to be checked on the router (so packet may not be tampered with).

A router acting as notary processes the packet this way:

- *Verifies the CGA of the emitter
- *Verifies the signature of the message (linked to CGA of the source address)
- *Verifies the CGA and signature of the inner packet
- *Responds with a Signature status message (defined in the following section)

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: inner packet CGA verification check failed
>
> 2: inner packet signature verification check failed
>
> 3: unsupported hash algorithm (to compute Hash1/Hash2)
>
> 4: unsupported Public Key algorithm
>
> 5: ask later (router is busy)

Request Identifier

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

This message is a response to a Notary signature check request message and is protected by SEND options generated using the public key contained in the certificate of the router authorized to act as notary. On reception of this message, a node performs CGA check and Universal Signature option check . Then, if the status message is 0, that node can now trust the original packet that created the need for a Notary signature check request message. This amounts to resuming the SEND protocol using secure packets. On a status value different from 0, the packet will be considered as unsecure and be treated as such.

---

7. Security Considerations

TOC

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, see 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),” Feb 2009.)).
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.
To be completed.

---

8. IANA Considerations

TOC

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

---

9. Acknowledgments

TOC

The authors gratefully acknowledge the contributions of Marcello Bagnulo-Braun, and other participants of the SEND working group.

---

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-00 (work in progress), Feb 2009 (TXT).

10.2. Informative References

TOC

[RFC4581]	Bagnulo, M. and J. Arkko, “Cryptographically Generated Addresses (CGA) Extension Field Format,” RFC 4581, October 2006 (TXT).
[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-02 (work in progress), November 2008 (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.

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