

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 12, 2012

A. Garcia-Martinez
M. Bagnulo
UC3M
April 10, 2012

Management Information Base for the SEcure Neighbor Discovery (SEND)
protocol
draft-garcia-martinez-sendmib-04

Abstract

This memo defines a portion of the Management Information Base (MIB) for managing the SEND (SEcure Neighbor Discovery) Protocol.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 12, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Internet-Draft

SEND MIB

April 2012

Table of Contents

1.	The Internet-Standard Management Framework	3
2.	Overview	3
3.	Conventions	5
4.	Definitions	5
5.	Security Considerations	35
6.	IANA Considerations	38
7.	Acknowledgements	39
8.	References	39
8.1.	Normative References	39
8.2.	Informative References	40
	Authors' Addresses	40

Internet-Draft

SEND MIB

April 2012

1. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to [section 7 of RFC 3410](#) [RFC3410]. Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, [RFC 2578](#) [RFC2578], STD 58, [RFC 2579](#) [RFC2579] and STD 58, [RFC 2580](#) [RFC2580].

2. Overview

This document defines SEND-MIB, the portion of the Management Information Base (MIB) to be used for managing the SEcure Neighbor Discovery (SEND) [RFC3971] protocol. SEND specifies security mechanisms to protect the Neighbor Discovery (ND) Protocol [RFC4861], [RFC4862], so it is IPv6-specific. To protect the ND protocol, it relies on the following elements

- o Certification Path, anchored on trusted parties, which can be used to certify the authority of the routers. A Certification Path can be exchanged among routers and hosts by means of two ICMP messages, the Certification Path Solicitation message and the Certification Path Advertisement message.
- o Cryptographically Generated Address (CGA) [RFC3972], an IPv6 address which can be used to prove address ownership
- o RSA Signature option, which protects ND messages by using keying material related to a Certification Path or a CGA.

SEND-MIB provides means to monitor and configure most of the elements related with SEND operation in hosts. In particular, it allows managing:

- o Authority attestation and verification. The following objects are

related to authority verification management in the receiver:

- * `sendCgaVerificationMinbits` and `sendCgaVerificationMaxbits` objects specify respectively the minimum and maximum number of bits that a key of a received CGA must have to be usable by SEND.
- * For each ND message type, a specific `sendRsaVerificationMethod` object indicates the method through which the authority of the remote node is determined (Certification Path and/or CGA).
- o Timestamp verification. The parameters that control the verification of the Timestamp option to provide anti-replay protection for unsolicited advertisements and redirects are managed through the discrete objects `sendTimestampDelta`,

`sendTimestampFuzz` and `sendTimestampDrift`.

- o Cryptographic information repository. Two tables store, after proper validation, the cryptographic information required by SEND:
 - * The `sendTrustAnchorTable` stores the trust anchors configured in the node. Read-create access to the objects of this table enables the use of network management protocols to create them.
 - * The `sendCertTable` stores the certificates received in Certification Path Advertisement messages. This information is stored as a linked list in order to reflect the dependencies of the certificates to a trust anchor or to other certificates of a Certification Path.
 - * The `sendCgaLocalTable` indicates which of the CGA addresses available in the node (shown in a non-SEND specific table defined in the CGA-MIB [[I-D.garcia-martinez-cgamib](#)]) can be used for SEND operation.
- o Compatibility with non-SEND devices. The `sendAcceptUnsecured` object manages the acceptance of unsecured ND messages in a system-wide basis. If unsecured ND messages are accepted, it is useful to know which of the information received is secured. To do so, three tables augment existing tables containing ND information [[RFC4293](#)] to indicate whether the information related to an address prefix, a default router, or an association between an IP address and a physical address has been secured by SEND. These tables are `sendIpAddressPrefixSecuredTable`, `sendIpDefaultRouterSecuredTable` and `sendIpNetToPhysicalSecuredTable`. Finally, `sendIgnoreUnsecuredDadFirstTentative` controls DAD operation in links shared with non-SEND devices.
- o SEND statistics. System-wide objects show statistics accounting

for secured and unsecured messages, and messages with validation errors, for both ND and Certification Path Advertisement messages.

Note that SEND-MIB depends on the implementation of the CGA-MIB [[I-D.garcia-martinez-cgamib](#)] for some of its objects, so it requires the CGA-MIB to exist in the managed system in which SEND-MIB is implemented. SEND-MIB also complements the information of IP-MIB tables[RFC4293].

SEND-MIB does not aim to manage SEND operation in routers, except for configurations common to both routers and hosts. The approach taken in this document is similar to that of IP-MIB [[RFC4293](#)], which does not provide objects to manage the generation of Router Advertisement messages, and in particular it does not provide means to manage prefix options. In a similar way, SEND-MIB does not provide means to manage the generation of Certification Path Advertisement, the security configuration of Router Advertisement messages, or the configuration of certification material in the routers. These configuration capabilities are left for future specifications.

[3.](#) Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

[4.](#) Definitions

SEND-MIB DEFINITIONS ::= BEGIN

IMPORTS

MODULE-IDENTITY, OBJECT-TYPE,	
Unsigned32, mib-2, TimeTicks,	
Integer32, Counter32	FROM SNMPv2-SMI
TEXTUAL-CONVENTION, TestAndIncr,	
RowStatus, StorageType, RowPointer	FROM SNMPv2-TC
MODULE-COMPLIANCE, OBJECT-GROUP	FROM SNMPv2-CONF
ipAddressPrefixEntry, ipDefaultRouterEntry,	
ipNetToPhysicalEntry	FROM IP-MIB
cgaLocalEntry	FROM CGA-MIB;

sendMIB MODULE-IDENTITY

LAST-UPDATED "201204090000Z"

ORGANIZATION "IETF CSI (Cga & Send Maintenance) Working Group"

CONTACT-INFO

"Editor:

Alberto Garcia-Martinez
U. Carlos III de Madrid
Avenida Universidad, 30
Leganes, Madrid 28911
Spain
Email: alberto.garcia@uc3m.es"

DESCRIPTION

"The MIB module for managing the SEND protocol.

Copyright (c) 2012 IETF Trust and the persons identified
as the document authors.
All rights reserved. This version of this MIB module is
part of RFC yyyy; see the RFC itself for full legal
notices."

-- RFC Ed.: replace yyyy with actual RFC number & remove this
-- note

Garcia-Martinez & Bagnulo Expires October 12, 2012

[Page 5]

Internet-Draft

SEND MIB

April 2012

REVISION "201204090000Z"

DESCRIPTION

"Initial version, published as RFC yyyy."

-- RFC Ed.: replace yyyy with actual RFC number & remove
-- this note

::= { mib-2 XXX }

-- RFC Ed.: replace XXX with actual number assigned by IANA &
-- remove this note

--

-- The textual conventions we define and use in this MIB.

--

SendRsaVerificationMethod ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Acceptable authorization method to validate the RSA signature option of a received NDP message.

In the trustAnchor(1) method, the option is verified according to the keying information obtained from a Certification Path rooted in a trust anchor known by both sender and receiver. The sender may claim additional authorization through the use of CGAs, but in this case it is neither required nor verified.

In the cga(2) method, the CGA property of the sender's address is verified. The sender may claim additional authority through a trust anchor, but in this case it is neither required nor verified.

In the trustAnchorAndCga(3) method, both the trust anchor and the CGA verification are required.

Finally, with the trustAnchorOrCga(4) method, either the trust anchor or the CGA verification is required."

REFERENCE "[RFC 3971](#) : [section 5.2.3](#)"

SYNTAX INTEGER {

trustAnchor(1),
cga(2),
trustAnchorAndCga(3),
trustAnchorOrCga(4)

}

SendCertificateIndex ::= TEXTUAL-CONVENTION

DISPLAY-HINT "d"

STATUS current

DESCRIPTION

"A unique value, greater than zero, for each certificate within a table.

The management station MUST select a pseudo-random number, so that this value could be used as an index. If this index was already in use, SO an 'inconsistentValue' was returned in response to the management protocol set

operation, the management station SHOULD select a new pseudo-random number and retry the operation.
The value for each certificate must remain constant at least from one re-initialization of the entity's network management system to the next re-initialization."

SYNTAX Unsigned32 (1..4294967295)

SendX509ASN1DEREncodedRouterAuthCert ::= TEXTUAL-CONVENTION

DISPLAY-HINT "4096x"

STATUS current

DESCRIPTION

"A Router Authorization Certificate, i.e. an X.509v3 certificate as defined in [\[RFC5280\]](#). It SHOULD contain at least one instance of the X.509 extension for IP addresses as defined in [\[RFC3279\]](#). The certificate is encoded as an ASN.1 DER [\[CCITT.X690.2002\]](#) object."

REFERENCE "[RFC 3971](#): 6.3.1, [RFC 5280](#), ITU-T Recommendation X.690"

SYNTAX OCTET STRING (SIZE (0..4096))

SendSecurityStatus ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A value that specifies whether the information associated to the entry in which an object of this type is defined has been secured by SEND or not."

SYNTAX INTEGER {
 sendSecured(1),
 sendNotSecured(2)
}

send OBJECT IDENTIFIER ::= { sendMIB 1 }

--

-- Operational status of SEND

--

--

-- Authority verification


```

sendCgaVerificationMinbits OBJECT-TYPE
    SYNTAX Integer32 (384 .. 2147483647)
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "Minimum acceptable key length for public keys used in the
        verification of remote CGA. The minimum RSA key length
        required for a CGA is 384 bits [RFC3972]."
```

REFERENCE "[RFC 3971](#) : [section 5.1.3](#)"

```

    DEFVAL { 1024 }
    ::= { send 1 }
```



```

sendCgaVerificationMaxbits OBJECT-TYPE
    SYNTAX Integer32 (384 .. 2147483647)
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "Maximum acceptable key length for public keys used in the
        verification of remote CGA. The value of this object can
        limit the amount of computation needed when verifying SEND
        messages.
        When set to 2147483647, it indicates that the node does
        not check for any maximum key length.
        This value MUST be equal or greater than
        sendCgaVerificationMinbits, and SHOULD be at least 2048
        bits [RFC3971]."
```

REFERENCE "[RFC 3971](#) : [section 5.1.3](#)"

```

    ::= { send 2 }
```



```

--
-- objects related with the authorization method used for verifying
-- RSA signature options for each NDP message type
--
```



```

sendRsaVerificationMethodNS OBJECT-TYPE
    SYNTAX SendRsaVerificationMethod
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "This object indicates the method through which the
        authority of the remote node is determined in the Neighbor
        Solicitation messages received.
        It affects all interfaces of the node."
```

REFERENCE "[RFC 3971](#) : [section 5.2.3](#)"

::= { send 3 }

sendRsaVerificationMethodNA OBJECT-TYPE

SYNTAX SendRsaVerificationMethod

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object indicates the method through which the authority of the remote node is determined in the Neighbor Advertisement messages received.

It affects all interfaces of the node."

REFERENCE "[RFC 3971](#) : [section 5.2.3](#)"

::= { send 4 }

sendRsaVerificationMethodRS OBJECT-TYPE

SYNTAX SendRsaVerificationMethod

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object indicates the method through which the authority of the remote node is determined in the Router Solicitation messages received.

It affects all interfaces of the node."

REFERENCE "[RFC 3971](#) : [section 5.2.3](#)"

::= { send 5 }

sendRsaVerificationMethodRA OBJECT-TYPE

SYNTAX SendRsaVerificationMethod

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object indicates the method through which the authority of the remote node is determined in the Router Advertisement messages received.

It affects all interfaces of the node."

REFERENCE "[RFC 3971](#) : [section 5.2.3](#)"

::= { send 6 }

sendRsaVerificationMethodRedirect OBJECT-TYPE

SYNTAX SendRsaVerificationMethod

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object indicates the method through which the authority of the remote node is determined in the Redirect messages received.

Internet-Draft

SEND MIB

April 2012

```
        It affects all interfaces of the node."
REFERENCE "RFC 3971 : section 5.2.3"
 ::= { send 7 }

--
-- objects associated to timestamp verification management
--

sendTimestampDelta OBJECT-TYPE
    SYNTAX Unsigned32
    UNITS "seconds"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "The maximum difference in absolute value between the
        timestamp included in a Neighbor Discovery message and the
        reception time of the message, measured in seconds with
        the local clock.
        It corresponds to the variable TIMESTAMP_DELTA defined in
        [RFC3971]."
```

```
REFERENCE "RFC 3971 : section 5.3.4.2, RFC 3971 : section 10.2"
DEFVAL { 300 }
 ::= { send 8 }

sendTimestampFuzz OBJECT-TYPE
    SYNTAX TimeTicks
    UNITS "milliseconds"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "Time in milliseconds, used to check upon the reception of
        a new SEND message if the following inequality holds:
 $TS_{new} + \text{sendTimestampFuzz} > TS_{last} + (RD_{new} - RD_{last}) \times (1 - \text{sendTimestampDrift}) - \text{sendTimestampFuzz}$ 
        With
        TSnew: timestamp of the newly received SEND message
        TSlast: timestamp of the previously received SEND message
        RDnew: reception time of the newly received SEND message
        RDlast: reception time of the previously received SEND
```

```
message.  
This object corresponds to the variable TIMESTAMP_FUZZ  
defined in [RFC3971]."  
REFERENCE "RFC 3971 : 5.3.4.2, RFC 3971 : section 10.2"  
DEFVAL { 100 }  
::= { send 9 }
```

sendTimestampDrift OBJECT-TYPE

```
SYNTAX Integer32 (0..10000)  
UNITS "0.01 Percentage"  
MAX-ACCESS read-write  
STATUS current  
DESCRIPTION  
    "The maximum clock drift allowed when validating the  
    timestamp of a received Neighbor Discovery message,  
    measured in parts per 10000 (or 0.01 percentage). It  
    corresponds to the variable TIMESTAMP_DRIFT defined in  
    [RFC3971]."  
REFERENCE "RFC 3971 : 5.3.4.2, RFC 3971 : section 10.2"  
DEFVAL { 100 }  
::= { send 10 }
```

```
--  
-- Cryptographic information repository  
--
```

```
--  
-- table of trust anchors known by a node  
--
```

sendTrustAnchorSpinLock OBJECT-TYPE

```
SYNTAX TestAndIncr  
MAX-ACCESS read-write  
STATUS current  
DESCRIPTION  
    "An advisory lock used to allow cooperating SNMP managers  
    to coordinate their use of the set operation in creating  
    or modifying rows within the sendTrustAnchorTable table.
```

In order to use this lock to coordinate the use of set operations, managers should first retrieve sendTrustAnchorSpinLock. They should then determine the appropriate row to create or modify, selecting a pseudo-random number for the sendTrustAnchorIndex. Finally, they should issue the appropriate set command including the retrieved value of sendTrustAnchorSpinLock. If another manager has altered the table in the meantime, then the value of sendTrustAnchorSpinLock will have changed and the creation will fail as it will be specifying an incorrect value for sendTrustAnchorSpinLock. It is suggested, but not required, that the sendTrustAnchorSpinLock be the first var bind for each set of objects representing a 'row' in a PDU."

::= { send 11 }

sendTrustAnchorTable OBJECT-TYPE

SYNTAX SEQUENCE OF SendTrustAnchorEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The list of trust anchors that are used for validating the authority of the Certification Path of a remote node. Entries can be created to configure new anchors. Anchor entries can also be deleted.

Note that the removal of entries in this table SHOULD result in the deletion of the certificates belonging to Certification Paths rooted in this anchor.

Note that if SEND has been enabled in a host, there SHOULD be at least one anchor."

REFERENCE "[RFC 3971](#) : 6.5"

::= { send 12 }

sendTrustAnchorEntry OBJECT-TYPE

SYNTAX SendTrustAnchorEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A trust anchor provided in the form of a self-signed

certificate.

When the value of the sendTrustAnchorStatus of an entry has been set once to enabled(2), the sendTrustAnchorCert MUST remain unmodified."

```
INDEX { sendTrustAnchorIndex }  
 ::= { sendTrustAnchorTable 1 }
```

```
SendTrustAnchorEntry ::= SEQUENCE {  
    sendTrustAnchorIndex SendCertificateIndex,  
    sendTrustAnchorCert SendX509ASN1DEREncodedRouterAuthCert,  
    sendTrustAnchorStatus INTEGER,  
    sendTrustAnchorRowStatus RowStatus,  
    sendTrustAnchorStorageType StorageType  
}
```

sendTrustAnchorIndex OBJECT-TYPE

SYNTAX SendCertificateIndex

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A unique value, greater than zero, for each trust anchor. The management station MUST select a pseudo-random number to use it as the index. If this index was already in use,

so an 'inconsistentValue' was returned in response to the management protocol set operation, the management station SHOULD select a new pseudo-random number and retry the operation.

The value of the sendTrustAnchorIndex for each trust anchor MUST remain constant at least from one re-initialization of the entity's network management system to the next re-initialization."

```
::= { sendTrustAnchorEntry 1 }
```

sendTrustAnchorCert OBJECT-TYPE

SYNTAX SendX509ASN1DEREncodedRouterAuthCert

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"Variable-length field containing the trust anchor in the form of a self-signed certificate.

Upon a set operation, an 'inconsistentValue' error MUST be

returned if the value is not a valid DER-encoded ASN.1 certificate.
This object MUST NOT be modified once the
cgaLocalRowStatus object has been set to enabled(2)."
 ::= { sendTrustAnchorEntry 2 }

sendTrustAnchorStatus OBJECT-TYPE

SYNTAX INTEGER {
 disabled(1),
 enabled(2) }

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This columnar object indicates whether the row can be used as a Trust Anchor in the managed system or not. When set to enabled(1), the administrator requires the trust anchor to be available for validating the authority of Certification Paths. Conversely, when set to disabled(2), the administrator requires the anchor not to be available for validating the authority of Certification Paths."

DEFVAL { disabled }

::= { sendTrustAnchorEntry 3 }

sendTrustAnchorRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The status of this conceptual row.
Once set to enabled, the value of sendTrustAnchorCert cannot be changed."

::= { sendTrustAnchorEntry 4 }

sendTrustAnchorStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The storage type for this conceptual row. If this object has a value of permanent(4), then no other objects are required to be able to be modified."

```

DEFVAL { volatile }
::= { sendTrustAnchorEntry 5 }

--
-- table of the certificates part of a Certification Path
-- received from a remote router
--

sendCertTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SendCertEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of standard Public Key Certificates (PKC, in the
        sense of [RFC5280]) which can be used by the SEND node.
        The managed element populates the entries in this table
        with the information being received from Certification
        Path Advertisement messages. Only verified certificates,
        i.e. certificates verified to the trust anchor or to a
        certificate that has been verified before (i.e. which
        validly belongs to a Certification Path), are included in
        the table. Certificates being certified by a trust anchor
        contain a RowPointer element that points to the parent
        entry in the sendTrustAnchorTable (note that there MAY be
        many certificates rooted to the same trust anchor).
        Certificates being signed by another certificate in the
        table contain a RowPointer element that points to the
        parent certificate in this table, to reflect the
        dependencies of the certificates forming a Certification
        Path.
        Entries in this table can be removed as a result of the
        normal operation of SEND. Additionally, the deletion of
        an anchor on which a Certification Path is rooted MUST

```

result in the deletion of all the certificates of the Certification Path.

In the period since the certificate has been verified to the time at which it is possible to check the information

obtained from the Certificate Revocation List, the sendCertStatus object indicates that the certificate is provisional(1).

The managed element can delete entries if the periodic Certificate Revocation List check fails (either in the provisional or in the valid states). In this case, all certificates depending on the revoked one MUST be deleted."

REFERENCE "[RFC 3971](#) : 6.4.2, [RFC 3971](#) : 6.4.6"

::= { send 13 }

sendCertEntry OBJECT-TYPE

SYNTAX SendCertEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Information related with a particular certificate."

INDEX { sendCertIndex }

::= { sendCertTable 1 }

SendCertEntry ::= SEQUENCE {

sendCertIndex SendCertificateIndex,

sendCertCert SendX509ASN1DEREncodedRouterAuthCert,

sendCertParentCertificate RowPointer,

sendCertStatus INTEGER

}

sendCertIndex OBJECT-TYPE

SYNTAX SendCertificateIndex

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A unique value, greater than zero, for each certificate. The management station MUST select a pseudo-random number to use it as index. If this index was already in use, so an 'inconsistentValue' was returned in response to the management protocol set operation, the management station SHOULD select a new pseudo-random number and retry the operation.

The value of the sendTrustAnchorIndex for each certificate MUST remain constant at least from one re-initialization of the entity's network management system to the next re-initialization."

```
 ::= { sendCertEntry 1 }

sendCertCert OBJECT-TYPE
    SYNTAX SendX509ASN1DEREncodedRouterAuthCert
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Variable-length field containing the certificate
        information."
    ::= { sendCertEntry 2 }

sendCertParentCertificate OBJECT-TYPE
    SYNTAX RowPointer
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "For the first certificate of a Certification Path, this
        object points to the row of the sendTrustAnchorTable of
        the anchor that certifies this certificate. For
        certificates other than the first of a Certification Path,
        the object points to the parent certificate entry in this
        table."
    ::= { sendCertEntry 3 }

sendCertStatus OBJECT-TYPE
    SYNTAX INTEGER {
        provisional(1),
        valid(2)
    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The status of the certificate.
        A certificate has a sendCertStatus value of
        provisional(1) if a Certificate Revocation List check has
        not been performed due to the lack of connection to
        internet. After validating the certificate in the CRL,
        sendCertStatus is changed to valid(2). Note that if
        subsequent checks to the CRL result in invalidation of the
        certificate, the certificate is removed from the table, so
        no state is required for this case."
    REFERENCE "RFC 3971 : section 6.3.1"
    ::= { sendCertEntry 4 }

--
-- CGA addresses configured in this node
--
```

Internet-Draft

SEND MIB

April 2012

```
sendCgaLocalTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SendCgaLocalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A table containing one column which indicates if each
        entry of CGA-MIB:cgaLocalTable can be used by SEND or not.
        An entry in the CGA-MIB:cgaLocalTable may not be used by
        SEND if, for example, it does not comply with the minimum
        key length specified by SEND. Note that other policies
        MAY be used to filter out configured CGA for SEND use."
    REFERENCE "[I-D.garcia-martinez-cgamib]"
    ::= { send 14 }

sendCgaLocalEntry OBJECT-TYPE
    SYNTAX SendCgaLocalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An entry of the sendCgaLocalTable.
        This table augments the CGA-MIB:cgaLocalTable, i.e. every
        entry in this table has a one-to-one correspondence with
        an entry in the CGA-MIB:cgaLocalTable."
    AUGMENTS { cgaLocalEntry }
    ::= { sendCgaLocalTable 1 }

SendCgaLocalEntry ::= SEQUENCE {
    sendCgaLocalStatus INTEGER
}

sendCgaLocalStatus OBJECT-TYPE
    SYNTAX INTEGER {
        validForSend(1),
        notValidForSend(2)
    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of this object indicates if its corresponding
        entry in CGA-MIB:cgaLocalTable can be used by SEND or not"
```

```
        when properly configured as an IP address."
 ::= { sendCgaLocalEntry 1 }

--
-- compatibility with non-SEND devices
--
```

sendAcceptUnsecured OBJECT-TYPE

```
SYNTAX INTEGER {
    acceptSecuredAndUnsecured(1),
    acceptOnlySecured(2)
}
MAX-ACCESS read-write
STATUS current
DESCRIPTION
    "This object indicates whether the node accepts or not
    unsecured messages.  When its value is
    acceptSecuredAndUnsecured(1), both secured and unsecured
    messages are processed, while when its value is
    acceptOnlySecured(2) unsecured messages are ignored.
    The acceptance of unsecured messages is allowed to ease
    transition from unsecured to secured links."
REFERENCE "RFC 3971 : section 8"
DEFVAL { acceptOnlySecured }
 ::= { send 15 }

--
-- Compatibility with non-SEND devices: security status of ND related
-- information
--

--
-- Augmentation of tables of [RFC4293] to indicate whether the
-- information included in an entry in the
-- IP-MIB:ipAddressPrefixTable, IP-MIB:ipDefaultRouterTable and
-- IP-MIB ipNetToPhysicalTable has been secured or not
--

-- augmentation of IP-MIB:ipAddressPrefixTable
```

sendIpAddressPrefixSecuredTable OBJECT-TYPE
 SYNTAX SEQUENCE OF SendIpAddressPrefixSecuredEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION
 "A table that contains one column indicating if each entry in the IP-MIB:ipAddressPrefixTable is secured or not. An entry in the IP-MIB:ipAddressPrefixTable is secured if the last message of the Router Advertisement message that caused the creation or last update of the entry was secured. An entry in IP-MIB:ipAddressPrefixTable is unsecured otherwise. This table contains one row for every address prefix entry on the node. This table augments the basic address prefix information of the IP-MIB:ipAddressPrefixEntry.

If an entry in the ipAddressPrefixTable is deleted, the corresponding entry in the sendIpAddressPrefixSecuredTable MUST be deleted."

REFERENCE "[RFC 3971: section 8](#), ipAddressPrefixEntry is defined in the IP-MIB [[RFC4293](#)]."
 ::= { send 16 }

sendIpAddressPrefixSecuredEntry OBJECT-TYPE
 SYNTAX SendIpAddressPrefixSecuredEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION
 "An entry of the sendIpAddressSecuredTable. Each row is for an address prefix.
 This table augments the IP-MIB:ipAddressPrefixTable, i.e., every entry in this table has a one-to-one correspondence with an entry in the IP-MIB:ipAddressPrefixTable."

AUGMENTS { ipAddressPrefixEntry }
 ::= { sendIpAddressPrefixSecuredTable 1 }

SendIpAddressPrefixSecuredEntry ::= SEQUENCE {
 sendIpAddressPrefixSecuredStatus SendSecurityStatus
 }

sendIpAddressPrefixSecuredStatus OBJECT-TYPE
 SYNTAX SendSecurityStatus

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of this object indicates whether its corresponding entry in the IP-MIB:ipAddressPrefixTable is secured (if the last Router Advertisement message that caused the creation or last update of the entry was secured by SEND) or not.

If the value of the object sendAcceptUnsecured is set to acceptOnlySecured(2), then the value of this object MUST contain sendSecured(1)."

::= { sendIpAddressPrefixSecuredEntry 1 }

-- augmentation of IP-MIB:ipDefaultRouterTable

sendIpDefaultRouterSecuredTable OBJECT-TYPE

SYNTAX SEQUENCE OF SendIpDefaultRouterSecuredEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A table that contains one column indicating whether each entry in the IP-MIB:ipDefaultRouterTable is secured or not. An entry in IP-MIB:ipDefaultRouterTable is secured if the last message of the Router Advertisement message that caused the creation or last update of the entry was secured. An entry in IP-MIB:ipDefaultRouterTable is unsecured otherwise. This table contains one row for every address prefix entry on the node. This table augments the basic address prefix information of the IP-MIB:ipDefaultRouterEntry.

If an entry in the ipDefaultRouterTable is deleted, the corresponding entry in the sendIpDefaultRouterSecuredTable MUST be deleted."

REFERENCE "[RFC 3971](#) : [section 8](#)

ipDefaultRouterEntry is defined in the IP-MIB [[RFC4293](#)]."

::= { send 17 }

```

sendIpDefaultRouterSecuredEntry OBJECT-TYPE
    SYNTAX SendIpDefaultRouterSecuredEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An entry of the sendIpDefaultRouterSecuredTable.  Each
        row is for a default router.
        This table augments the IP-MIB:ipDefaultRouterTable, i.e.,
        every entry in this table has a one-to-one correspondence
        with an entry in the IP-MIB:ipDefaultRouterTable."
    AUGMENTS { ipDefaultRouterEntry }
    ::= { sendIpDefaultRouterSecuredTable 1}

SendIpDefaultRouterSecuredEntry ::= SEQUENCE {
    sendIpDefaultRouterSecuredStatus SendSecurityStatus
}

sendIpDefaultRouterSecuredStatus OBJECT-TYPE
    SYNTAX SendSecurityStatus
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of this object indicates whether its
        corresponding entry in the IP-MIB:ipDefaultRouterTable is
        secured (if the last Router Advertisement message that
        caused the creation or last update of the entry was
        secured by SEND) or not.

```

```

        If the value of the object sendAcceptUnsecured is set to
        acceptOnlySecured(2), then the value of this object MUST
        contain sendSecured(1)."
    ::= { sendIpDefaultRouterSecuredEntry 1 }

-- augmentation of IP-MIB:ipNetToPhysicalTable

sendIpNetToPhysicalSecuredTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SendIpNetToPhysicalSecuredEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION

```

"A table that contains one column indicating if each entry in the IP-MIB:ipNetToPhysicalTable is secured or not. An entry in IP-MIB:ipNetToPhysicalTable is secured if the last ND message that caused the creation or last update of the entry was secured. An entry in IP-MIB:ipNetToPhysicalEntry is unsecured otherwise. This table contains one row for every address prefix entry on the node. This table augments the basic address prefix information of the IP-MIB:ipNetToPhysicalEntry. If an entry in the IP-MIB:ipNetToPhysicalTable is deleted, the corresponding entry in the sendIpNetToPhysicalSecuredTable MUST be deleted."

REFERENCE "[RFC 3971](#) : [section 8](#)

ipNetToPhysicalEntry is defined in the IP-MIB [[RFC4293](#)]."

::= { send 18 }

sendIpNetToPhysicalSecuredEntry OBJECT-TYPE

SYNTAX SendIpNetToPhysicalSecuredEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry of the sendIpNetToPhysicalSecuredTable. Each row is for a neighbor.

This table augments the IP-MIB:ipNetToPhysicalTable, i.e., every entry in this table has a one-to-one correspondence with an entry in the IP-MIB:ipNetToPhysicalTable.

If the value of sendIpNetToPhysicalSecuredStatus for an entry is sendSecured(1), and the CGA has been used for validating this information, an entry in CGA-

MIB:cgaRemoteTable MUST exist containing the CGA parameter data structure of the CGA of the neighbor node."

AUGMENTS { ipNetToPhysicalEntry }

::= { sendIpNetToPhysicalSecuredTable 1 }

SendIpNetToPhysicalSecuredEntry ::= SEQUENCE {

```

        sendIpNetToPhysicalSecuredStatus SendSecurityStatus
    }

```

sendIpNetToPhysicalSecuredStatus OBJECT-TYPE

SYNTAX SendSecurityStatus


```

MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "The value of this object indicates whether its
    corresponding entry in the IP-MIB:ipNetToPhysicalTable is
    secured (if the last ND message that caused the creation
    or last update of the entry was secured by SEND).
    If the value of the object sendAcceptUnsecured is set to
    acceptOnlySecured(2), then the value of this object MUST
    contain sendSecured(1)."
```

::= { sendIpNetToPhysicalSecuredEntry 1 }

```

sendIgnoreUnsecuredDadFirstTentative OBJECT-TYPE
    SYNTAX INTEGER {
        acceptUnsecuredDadFirstTentative(1),
        ignoreUnsecuredDadFirstTentative(2)
    }
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "This object specifies whether the node accepts or ignores
        unsecured advertisements received when performing
        Duplicate Address Detection for the first CGA tentative
        address. sendIgnoreUnsecuredDadFirstTentative can be
        configured to ignoreUnsecuredDadFirstTentative(2) as a
        recovery mechanism for cases in which attacks against the
        first address become common. Note that for the second and
        third tentative addresses, only secured advertisements are
        considered."
    REFERENCE "RFC 3971 : section 8"
    DEFVAL { acceptUnsecuredDadFirstTentative }
    ::= { send 19 }
```

```

--
-- SEND statistics
--
```

```

sendStatsInNDMsgs OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
```

STATUS current

DESCRIPTION

"The total number of ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) that the node received. Note that this counter includes all the messages counted by sendStatsInNDSecuredWithErrors."

::= { send 20 }

sendStatsInNDSecuredValidMsgs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) that were both secured and successfully validated according to the rules specified in [[RFC3971](#)]."

REFERENCE "[RFC 3971](#) : 5.1.2, [RFC 3971](#) : 5.2.2"

::= { send 21 }

sendStatsInNDSecuredInvalidMsgs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) that contained any kind of SEND security information, but were not successfully validated according to the rules specified in [[RFC3971](#)]. Note that these messages MAY be further processed by the node as unsecured messages (and counted in sendStatsInNDUnsecuredMsgs), depending on the configuration of the node."

REFERENCE "[RFC 3971](#) : 5.1.2, [RFC 3971](#) : 5.2.2"

::= { send 22 }

sendStatsInNDUnsecuredMsgs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) processed as unsecured. It includes messages that did not contain any security

Internet-Draft

SEND MIB

April 2012

information defined in SEND, and messages containing security information for which validation failed and were finally processed as unsecured."

```
::= { send 23 }
```

sendStatsInNDSecuredWithErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) received by the node which were validated as secured but had ICMP-specific errors (bad ICMP checksums, bad length, bad format, etc.). Note that it does not include the messages counted by sendStatsInNDUnsecuredMsgs."

```
::= { send 24 }
```

sendStatsOutNDMsgs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) that the node attempted to send. This counter includes all messages counted by sendStatsOutNDSecuredErrors."

```
::= { send 25 }
```

sendStatsOutNDSecuredMsgs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of secured ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) attempted to be sent by the node."

REFERENCE "[RFC 3971](#) : 5.1.2, [RFC 3971](#) : 5.2.2"

```
::= { send 26 }
```

sendStatsOutNDSecuredErrors OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only

Internet-Draft

SEND MIB

April 2012

STATUS current
DESCRIPTION

"The number of secured ND messages (Neighbor Solicitation, Neighbor Advertisement, Router Solicitation, Router Advertisements and Redirects) attempted to be sent by the node that was not sent due to problems discovered within ICMP, such as a lack of buffers. This value SHOULD NOT include errors discovered outside the ICMP layer, such as the inability of IP to route the resultant datagram. In some implementations, there may be no types of error that contribute to this counter's value."

::= { send 27 }

sendStatsInCertMsgs OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The number of Certification Path Advertisement messages received by the node."

::= { send 28 }

sendStatsInCertVerifiedMsgs OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The number of Certification Path Advertisement messages received by a node which could be verified according to the verification procedure defined in [section 6.3 of \[RFC3971\]](#)."

REFERENCE "[RFC 3971](#) : [section 6.3](#)"

::= { send 29 }

sendStatsInCertInvalidMsgs OBJECT-TYPE

```

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "The number of Certification Path Advertisement messages
    received by a host which it could not verify successfully,
    according to the verification procedure defined in section
    6.3 of \[RFC3971\]. It does not include messages counted by
    sendStatsInCertErrors."
REFERENCE "RFC 3971 : section 6.3"
 ::= { send 30 }

```

sendStatsInCertErrors OBJECT-TYPE

```

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "The number of Certification Path Advertisement messages
    received that had ICMP-specific errors (bad ICMP
    checksums, bad length, bad format, etc.). Note that it
    does not include the messages counted by
    sendStatsInCertFailedMessages."
 ::= { send 31 }

```

```

--
-- conformance information
--

```

```

sendMIBConformance OBJECT IDENTIFIER ::= { sendMIB 2 }

```

```

sendMIBCompliances OBJECT IDENTIFIER ::= { sendMIBConformance 1 }

```

```

sendMIBGroups OBJECT IDENTIFIER ::= { sendMIBConformance 2 }

```

```

-- compliance statement #1

```

```

sendMIBCompatibleCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION

```

```

        "Compliance statement for a managed element that supports
        compatible operation with non-SEND devices, and it is
        fully configurable (read-create access for the
        sendTrustAnchorGroup, and read-write access for all
        writable objects)."
```

MODULE -- this module

```

MANDATORY-GROUPS {
    sendAuthorityGroup, sendTimestampGroup,
    sendTrustAnchorGroup, sendCertGroup,
    sendCgaLocalGroup, sendCompatibilityGroup,
    sendStatsGroup }

OBJECT sendTrustAnchorRowStatus
SYNTAX RowStatus { active(1), notInService (2) }
WRITE-SYNTAX RowStatus { active(1), notInService (2),
    createAndGo(4), destroy(6) }

DESCRIPTION
    "Support for createAndWait is not required."
```

```

OBJECT sendCgaVerificationMaxbits
MIN-ACCESS read-only
DESCRIPTION
    "An agent is not required to provide write access to this
    object, since this object MAY not be configurable."

 ::= { sendMIBCompliances 1 }

-- compliance statement #2

sendMIBSendOnlyCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
    "Compliance statement for an managed element that does not
    support compatible operation with non-SEND devices and it
    is fully configurable (read-create access for the
    sendTrustAnchorGroup, and read-write access for all
    writable objects)."
```

MODULE -- this module

```

MANDATORY-GROUPS {
```

```
sendAuthorityGroup, sendTimestampGroup,  
sendTrustAnchorGroup, sendCertGroup,  
sendCgaLocalGroup, sendStatsGroup }
```

```
OBJECT sendTrustAnchorRowStatus
```

```
SYNTAX RowStatus { active(1), notInService (2) }
```

```
WRITE-SYNTAX RowStatus { active(1), notInService (2),  
createAndGo(4), destroy(6) }
```

```
DESCRIPTION
```

```
"Support for createAndWait is not required."
```

```
OBJECT sendCgaVerificationMaxbits
```

```
MIN-ACCESS read-only
```

```
DESCRIPTION
```

```
"An agent is not required to provide write access to this  
object, since this object MAY not be configurable."
```

```
::= { sendMIBCompliances 2 }
```

```
-- compliance statement #3
```

```
sendMIBCompatibleReadOnlyCompliance MODULE-COMPLIANCE
```

```
STATUS current
```

```
DESCRIPTION
```

```
"Compliance statement for a managed element that supports  
compatible operation with non-SEND devices, but it is  
implemented without support for the creation of rows in
```

```
the sendTrustAnchorGroup, and with read-only access for  
the remaining writable objects."
```

```
MODULE -- this module
```

```
MANDATORY-GROUPS {
```

```
sendAuthorityGroup, sendTimestampGroup,  
sendTrustAnchorGroup, sendCertGroup,  
sendCgaLocalGroup, sendCompatibilityGroup,  
sendStatsGroup }
```

```
OBJECT sendCgaVerificationMinbits
```

```
MIN-ACCESS read-only
```

```
DESCRIPTION
```

"An agent is not required to provide write access to this object."

OBJECT sendCgaVerificationMaxbits

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodNS

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodNA

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodRS

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodRA

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodRedirect

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTimestampDelta

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTimestampFuzz

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTimestampDrift

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTrustAnchorSpinLock

MIN-ACCESS not-accessible

DESCRIPTION

"An agent is not required to implement this object. However, if an agent provides write access to any of the other objects in the sendTrustAnchorGroup, it SHOULD provide write access to this object as well."

OBJECT sendTrustAnchorCert

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTrustAnchorStatus

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write or create access to this object."

OBJECT sendTrustAnchorRowStatus

```

SYNTAX RowStatus { active(1) }
MIN-ACCESS read-only
DESCRIPTION
    "An agent is not required to provide write or create
    access to this object. In this case, the only value
    permitted is active(1)."
```

OBJECT sendTrustAnchorStorageType

```

MIN-ACCESS read-only
DESCRIPTION
    "An agent is not required to provide write or create
    access to this object. If an agent allows this object to
    be written or created, it is not required to allow this
    object to be set to nonVolatile(3), permanent(4) or
    readOnly(5)."
```

OBJECT sendAcceptUnsecured

```

MIN-ACCESS read-only
DESCRIPTION
    "An agent is not required to provide write access to this
    object."
```

OBJECT sendIgnoreUnsecuredDadFirstTentative

```

MIN-ACCESS read-only
DESCRIPTION
    "An agent is not required to provide write access to this
    object."
```

```

::= { sendMIBCompliances 3 }
```

-- compliance statement #4

```

sendMIBSendOnlyReadOnlyCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
    "Compliance statement for an agent that does not support
    compatible operation with non-SEND devices and it is
    implemented without support for read-create (i.e., in
    read-only mode)."
```

MODULE -- this module

```

MANDATORY-GROUPS { sendAuthorityGroup, sendTimestampGroup,
    sendTrustAnchorGroup, sendCertGroup,
    sendCgaLocalGroup, sendStatsGroup }
```

Internet-Draft

SEND MIB

April 2012

OBJECT sendCgaVerificationMinbits

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendCgaVerificationMaxbits

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodNS

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodNA

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodRS

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodRA

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendRsaVerificationMethodRedirect

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTimestampDelta

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTimestampFuzz

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTimestampDrift

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTrustAnchorSpinLock

MIN-ACCESS not-accessible

DESCRIPTION

"An agent is not required to implement this object. However, if an agent provides write access to any of the other objects in the sendTrustAnchorGroup, it SHOULD provide write access to this object as well."

OBJECT sendTrustAnchorCert

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write access to this object."

OBJECT sendTrustAnchorRowStatus

SYNTAX RowStatus { active(1) }

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write or create access to this object."

OBJECT sendTrustAnchorStorageType

MIN-ACCESS read-only

DESCRIPTION

"An agent is not required to provide write or create access to this object. If an agent allows this object to be written or created, it is not required to allow this object to be set to nonVolatile(3), permanent(4) or readOnly(5)."

::= { sendMIBCompliances 4 }

-- Units of conformance

sendAuthorityGroup OBJECT-GROUP

OBJECTS {

sendCgaVerificationMinbits,
sendCgaVerificationMaxbits,
sendRsaVerificationMethodNS,
sendRsaVerificationMethodNA,
sendRsaVerificationMethodRS,
sendRsaVerificationMethodRA,
sendRsaVerificationMethodRedirect
}

STATUS current

DESCRIPTION

"The group of objects that control SEND authority attestation and verification processes. These objects manage the acceptable size of the CGAs being used and the checks that are applied to incoming SEND authority information."

::= { sendMIBGroups 1 }

sendTimestampGroup OBJECT-GROUP

OBJECTS {

sendTimestampDelta, sendTimestampFuzz, sendTimestampDrift
}

STATUS current

DESCRIPTION

"The group of objects that control the verification of the Timestamp option to provide anti-replay protection"

::= { sendMIBGroups 2 }

sendTrustAnchorGroup OBJECT-GROUP

```

OBJECTS {
    sendTrustAnchorSpinLock,
    sendTrustAnchorCert,
    sendTrustAnchorStatus,
    sendTrustAnchorRowStatus,
    sendTrustAnchorStorageType
}
STATUS current
DESCRIPTION
    "The group of objects for managing the anchors used for
    validating the authority of the Certification Path of a
    remote node."
::= { sendMIBGroups 3 }

sendCertGroup OBJECT-GROUP
OBJECTS {
    sendCertCert,

```

```

    sendCertParentCertificate,
    sendCertStatus
}
STATUS current
DESCRIPTION
    "The group of objects for storing the certificates that
    belong to a Certification Path received from a remote
    router."
::= { sendMIBGroups 4 }

sendCgaLocalGroup OBJECT-GROUP
OBJECTS {
    sendCgaLocalStatus
}
STATUS current
DESCRIPTION
    "The group for the object pointing to the CGA which can be
    used by SEND as local addresses."
::= { sendMIBGroups 5 }

sendCompatibilityGroup OBJECT-GROUP
OBJECTS { sendAcceptUnsecured, sendIpAddressPrefixSecuredStatus,

```

```

sendIpDefaultRouterSecuredStatus,
sendIpNetToPhysicalSecuredStatus,
sendIgnoreUnsecuredDadFirstTentative }
STATUS current
DESCRIPTION
    "The group of objects to configure SEND operation to
    interact with non-SEND hosts and to show the security
    status of the resulting ND information."
 ::= { sendMIBGroups 6 }

```

```

sendStatsGroup OBJECT-GROUP
OBJECTS {
    sendStatsInNDMsgs,
    sendStatsInNDSecuredValidMsgs,
    sendStatsInNDSecuredInvalidMsgs,
    sendStatsInNDUnsecuredMsgs,
    sendStatsInNDSecuredWithErrors,
    sendStatsOutNDMsgs,
    sendStatsOutNDSecuredMsgs,
    sendStatsOutNDSecuredErrors,
    sendStatsInCertMsgs,
    sendStatsInCertVerifiedMsgs,
    sendStatsInCertInvalidMsgs,
    sendStatsInCertErrors
}
STATUS current
DESCRIPTION
    "The group of statistics related to SEND operation."
 ::= { sendMIBGroups 7 }

```

```

    }
STATUS current
DESCRIPTION
    "The group of statistics related to SEND operation."
 ::= { sendMIBGroups 7 }

```

END

5. Security Considerations

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write or read-create. Such objects may be considered sensitive or vulnerable in some network environments. In the following paragraphs we discuss the risks

- associated to improper SET access to the objects of the MIB module.
- o `sendCgaVerificationMinbits`. Reducing the value of this object may facilitate the impersonation of a legitimate CGA by an attacker: the time to generate a key pair can be shortened, so the time to perform a brute force attack to find a key pair with the same hash as the attacked CGA can be shortened. On the other hand, incrementing the value of `sendCgaVerificationMinbits` may result in a DOS attack when only secured ND messages are accepted by the node, since messages from other nodes could be unintendedly discarded if having a CGA with a key shorter than `sendCgaVerificationMinbits`.
 - o `sendCgaVerificationMaxbits`. On one hand, low values of the `sendCgaVerificationMaxbits` object can result in discarding messages secured with a CGA larger than the limit established in this object, resulting in a DOS attack. On the other hand, an attacker could set high values to defeat the purpose of this configurable parameter, i.e. protect against the high resource consumption of resources in the receiver of a signed message.
 - o `sendRsaVerificationMethodNS`, `sendRsaVerificationMethodNA`, `sendRsaVerificationMethodRS`, `sendRsaVerificationMethodRA` and `sendRsaVerificationMethodRedirect`. The modification of the authorization method used for validating any of the messages of the ND protocol by an attacker can result in a DOS attack. The DOS attack occurs when the node is configured to require a type of authorization that it is not compatible with the authorization being used by the party generating the message.
 - o `sendTimestampDelta`, `sendTimeFuzz` and `sendTimestampDrift`. Incrementing the value of the objects `sendTimestampDelta`, `sendTimeFuzz` and `sendTimestampDrift` extends the vulnerability window for replay attacks. [Section 9.2.5 of \[RFC3971\]](#) discusses the impact of this action. On the other hand, setting too low values to these three objects may result in a DOS attack, since valid SEND messages could be discarded by too tight validation

- constraints.
- o `sendTrustAnchorTable`. The addition of an entry in the anchor table by an unauthorized party can be used to make the node trust (in SEND sense) in the information contained in Router Advertisements issued by a malicious node in the link. For example, this malicious node can become for the attacked node a default router (and perform MITM attacks for off-link destinations), it can alter the list of prefixes used for address

autoconfiguration, and the list of prefixes considered to be on-link. The unauthorized disabling or deletion of an entry in the sendTrustAnchorTable can result in DOS for the attacked node, since the node will not consider validly secured information as legitimate.

- o sendAcceptUnsecured. Write access from an unauthorized party to unintentionally accept unsecured information can be used to enable an attacker to introduce unprotected information into the ND data structures. Note that the possible attacks are limited by the fact that in general secured information is preferred to unsecured, as described by [\[RFC3971\] section 8](#). Conversely, if the value of the sendAcceptUnsecured object is unintentionally changed from acceptSecuredAndUnsecured to acceptOnlySecured, the node will lose the communication with non-SEND devices in the link considered, being this a DOS attack.
- o sendIgnoreUnsecuredDadFirstTentative. As [\[RFC3971\], section 9.2.3](#) states, if this object value is maliciously changed to ignore unsecured Neighbor Advertisements when DAD is being performed, a potential address collision between a SEND node and a non-SEND node may occur. The probability and effects of such an address collision are discussed in [\[RFC3972\]](#). On the other side, if the node is maliciously changed to accept unsecured Neighbor Advertisements as response to DAD, a node controlled by an attacker can force the generation of a second CGA. Since the DAD procedure in a SEND node for the second CGA only considers secured responses, regardless the state of this object, this impact is low.

The risk associated to unintended access to the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) are now discussed.

- o sendCgaVerificationMinbits. Read access to this object by an unauthorized node can provide information about the minimum key length to use in a brute force attack to impersonate the CGA of other node. If this value is not known by the attacker, the attacker could use in a brute force attack keys with a length shorter than the minimum acceptable, and so being unable to obtain valid values for the CGA Parameter Data Structure. Note that the attacker could blindly estimate a reasonable key length (such as the suggested default value, 1024 bits) to perform this attack

with high probability of success without knowing this value.

- o sendCgaVerificationMaxbits. Read access to this object by an unauthorized node can provide information about the maximum key length that can be claimed to try to drain CPU resources forcing the managed system to compute useless key verifications.
- o sendTimestampDelta, sendTimeFuzz and sendTimestampDrift. Knowing these values can make an attacker aware of the anti-replay protection window. However, if the values are set appropriately, the risks are low, as described in [Section 9.2.5 of \[RFC3971\]](#).
- o sendTrustAnchorTable. Having read access to the rows in the sendTrustAnchorTable allows a node to know which are the trust anchors in which the managed node rely on. An attacker could focus on breaking any of these keys to perform further attacks to the node. However, it is very likely that this information can also be inferred from the certificates being exchanged in the link to which the managed node is connected. The attacker can ask directly the router with a CPS message to obtain this information.
- o sendCertTable. Accessing to the sendCertTable does not reveal any information which could not be obtained by the attacker from a CPS/CPA message exchange. Therefore accessing to this information is not sensitive for the security of the managed node.
- o sendCgaLocalTable. Accessing to this information is not sensitive for the CGA configured in the link to which the attacker is also connected, since the attacker can obtain the same information by issuing a NS message to the managed node (or by capturing messages exchanged among the node and other nodes). The table can reveal CGAs configured in other links. Note that access to the IP-MIB:ipAddressTable also reveals other addresses configured in the same node (even though it cannot be determined if they are CGA or not). The risks of accessing to CGA-specific information contained in the CGA-MIB:cgaLocalTable are discussed in [\[I-D.garcia-martinez-cgamib\]](#).
- o sendAcceptUnsecured. An attacker could know by accessing to this information if the managed node is willing to accept unsecured messages. It can also try to influence in the managed node ND database by issuing unsecured messages, even without knowing the value of this object. Note that the attacker could obtain the same information by exchanging unsecured ND information with the node to test if the node accepts it.
- o sendIpAddressPrefixSecuredTable, sendIpDefaultRouterSecuredTable, sendIpNetToPhysicalSecuredTable. An attacker could know, by reading this tables, which of the prefixes, default routers and neighbor information have been introduced securely in the managed node as a result of SEND operation. This can be used, for example, to try to modify the information acquired as a result of an unsecured ND exchange. This does not provide much advantage for the attacker over just trying to introduce this information by sending ND messages to the managed node, or by identifying by

- traffic inspection or solicitation messages which routers or other node present in the link use SEND and which use plain ND.
- o `sendIgnoreUnsecuredDadFirstTentative`. An attacker accessing to this value could know in advance if sending unsecured NA when the managed node performs DAD will result in the managed node discarding the first address tried. Note that the impact of this attack is low, as discussed in [[RFC3972](#)], and this attack could only be exercised the next time the managed node tries to configure a new CGA.
 - o Objects collecting statistics about SEND operation. The disclosure of the information available in these objects affects mainly to privacy.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPsec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [[RFC3410](#)], [section 8](#)), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

[6.](#) IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
-----	-----
send-MIB	{ mib-2 XXX }

Editor's Note (to be removed prior to publication): the IANA is

requested to assign a value for "XXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the

Internet-Draft

SEND MIB

April 2012

assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

7. Acknowledgements

The work of Alberto Garcia-Martinez was supported in part by T2C2 project (TIN2008-06739-C04-01, granted by the Spanish Science and Innovation Ministry).

The authors would like to thank Suresh Krishnan for reviewing the document.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, [RFC 2578](#), April 1999.
- [RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, [RFC 2579](#), April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, [RFC 2580](#), April 1999.
- [RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 3279](#), April 2002.

- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 5280](#), May 2008.
- [RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", [RFC 3971](#), March 2005.
- [RFC4293] Routhier, S., "Management Information Base for the

Internet Protocol (IP)", [RFC 4293](#), April 2006.

[CCITT.X690.2002]

International International Telephone and Telegraph Consultative Committee, "ASN.1 encoding rules: Specification of basic encoding Rules (BER), Canonical encoding rules (CER) and Distinguished encoding rules (DER)", CCITT Recommendation X.690, July 2002.

[I-D.garcia-martinez-cgamib]

Garcia-Martinez, A. and M. Bagnulo, "Management Information Base for Cryptographically Generated Addresses (CGA)", [draft-garcia-martinez-cgamib-04](#) (work in progress), April 2012.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", [RFC 3972](#), March 2005.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", [RFC 4861](#), September 2007.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", [RFC 4862](#), September 2007.

[8.2.](#) Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", [RFC 3410](#), December 2002.

Authors' Addresses

Alberto Garcia-Martinez
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6249500
Email: alberto@it.uc3m.es
URI: <http://www.it.uc3m.es>

Garcia-Martinez & Bagnulo Expires October 12, 2012

[Page 40]

Internet-Draft

SEND MIB

April 2012

Marcelo Bagnulo
U. Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
Spain

Phone: +34 91 6248814
Email: marcelo@it.uc3m.es
URI: <http://www.it.uc3m.es/>

