SEcure Neighbor Discovery (SEND)
Source Address Validation Improvement (SAVI)

Abstract

This memo specifies SEcure Neighbor Discovery (SEND) Source Address
Validation Improvement (SAVI), a mechanism to provide source address
validation using the SEND protocol. The proposed mechanism
complements ingress filtering techniques to provide a finer
granularity on the control of IPv6 source addresses.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force
(IETF). It represents the consensus of the IETF community. It has
received public review and has been approved for publication by the
Internet Engineering Steering Group (IESG). Further information on
Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata,
and how to provide feedback on it may be obtained at
http://www.rfc-editor.org/info/rfc7219.

Copyright Notice

Table of Contents

1. Introduction

   This memo specifies SEND SAVI, a mechanism to provide source address
   validation for IPv6 networks using the SEND protocol [RFC3971]. The
   proposed mechanism complements ingress filtering techniques to
   provide a finer granularity on the control of the source addresses
   used.

   SEND SAVI uses the DAD_NSOL (Duplicate Address Detection Neighbor
   SOLicitation) and the DAD_NADV (DAD Neighbor ADVertisement) messages
   defined in [RFC4862] and the NUD_NSOL (Neighbor Unreachability
   Detection Neighbor SOLicitation) and NUD_NADV (NUD Neighbor
   ADVertisement) messages defined in [RFC4861] to validate the address
   ownership claim of a node. Using the information contained in these
   messages, host IPv6 addresses are associated to switch ports, so that
   data packets will be validated by checking for consistency in this
   binding, as described in [RFC7039]. In addition, SEND SAVI prevents
   hosts from generating packets containing off-link IPv6 source
   addresses.

   Scalability of a distributed SAVI system comprising multiple SEND
   SAVI devices is preserved by means of a deployment scenario in which
   SEND SAVI devices form a "protection perimeter". In this deployment
   scenario, the distributed SAVI system only validates the packets when
   they ingress to the protection perimeter, not in every SEND SAVI
   device traversed.

   The SEND SAVI specification, as defined in this document, is limited
   to links and prefixes in which every IPv6 host and every IPv6 router
   uses the SEND protocol [RFC3971] to protect the exchange of Neighbor
   Discovery information. If the SEND protocol is not used, we can
   deploy other SAVI solutions relying on monitoring different address
   configuration mechanisms to prove address ownership. For example,
   FCFS (First-Come, First-Served) SAVI [RFC6620] can be used by nodes
   locally configuring IPv6 addresses by means of the Stateless Address
   Autoconfiguration mechanism [RFC4862].

   SEND SAVI is designed to be deployed in SEND networks with as few
   changes to the deployed implementations as possible. In particular,
   SEND SAVI does not require any changes in the nodes whose source
   address is to be verified. This is because verification solely
   relies in the usage of already available protocols. Therefore, SEND
   SAVI neither defines a new protocol nor defines any new message on
   existing protocols, nor does it require that a host or router use an
   existing protocol message in a different way.

   An overview of the general framework about Source Address Validation
   Improvement is presented in [RFC7039].

1.1. Requirements Language

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 [RFC2119].

2. Background on SEND SAVI

2.1. Address Validation Scope

The application scenario of SEND SAVI is limited to the local link.
This means that the goal of SEND SAVI is to verify that the source
addresses of the packets generated by the nodes attached to the local
link have not been spoofed and that only legitimate routers generate
packets with off-link IPv6 source addresses.

In a link, there usually are hosts and routers attached. Hosts
generate packets with their own addresses as the source address.
This is called "local traffic". Routers may send packets containing
a source address other than their own, since they can forward packets
generated by other hosts (usually located in a different link). This
is the so-called transit traffic.

SEND SAVI allows the validation of the source address of the local
traffic, i.e., it allows verification that the source addresses of
the packets generated by the nodes attached to the local link have
not been spoofed. SEND SAVI also provides means to prevent hosts
from generating packets with source addresses derived from off-link
prefixes. However, SEND SAVI does not provide the means to verify if
a given router is actually authorized to forward packets containing a
particular off-link source address. Other techniques, like ingress
filtering [RFC2827], are recommended to validate transit traffic.

2.2. Binding Creation for SEND SAVI

SEND SAVI devices filter packets according to bindings between a
layer-2 anchor (the binding anchor) and an IPv6 address. These
bindings should allow legitimate nodes to use the bounded IPv6
address as source address and prevent illegitimate nodes from doing
so.

Any SAVI solution is not stronger than the binding anchor it uses.
If the binding anchor is easily spoofable (e.g., a Media Access
Control (MAC) address), then the resulting solution will be weak.
The treatment of non-compliant packets needs to be tuned accordingly.
In particular, if the binding anchor is easily spoofable and the SEND
SAVI device is configured to drop non-compliant packets, then the
usage of SEND SAVI may open a new vector of Denial-of-Service (DoS)

attacks, based on spoofed binding anchors. For that reason,
implementations of this specification use switch ports as their
binding anchors. Other forms of binding anchors are out of the scope
of this specification, and proper analysis of the implications of
using them should be performed before their usage.

SEND [RFC3971] provides tools to assure that a Neighbor Discovery
(ND) message containing a Cryptographically Generated Address (CGA)
[RFC3972] option and signed by an RSA option has been generated by
the legitimate owner of the CGA IPv6 address.

SEND SAVI uses SEND-validated messages to create bindings between the
CGA and the port of the SEND SAVI device from which it is reasonable
to receive packets with the CGA as the source address. The events
that trigger the binding creation process in a SEND SAVI device are:

o The reception of a DAD_NSOL message, indicating the attempt of a
   node to configure an address. This may occur when a node
   configures an address for the first time or after being idle for
   some time or when the node has changed the physical attachment
   point to the layer-2 infrastructure.

o The reception of any other packet (including data packets) with a
   source address for which no binding exists. This may occur if
   DAD_NSOL messages were lost, a node has changed the physical
   attachment point to the layer-2 infrastructure without issuing a
   DAD_NSOL message, a SAVI device loses a binding (for example, due
   to a restart), or the link topology changed.

When the binding creation process is triggered, the SEND SAVI device
has to assure that the node for which the binding is to be created is
the legitimate owner of the address. For the case in which the
binding creation process is initiated by a DAD_NSOL exchange, the
SEND SAVI device waits for the reception of a validated DAD_NADV
message, indicating that the other node has configured the address
before, or validated DAD_NSOL messages arriving from other locations,
indicating that another node is trying to configure the same address
at the same time. For the case in which packets other than a
DAD_NSOL initiate the creation of the binding, the SEND SAVI device
explicitly requires the node sending those packets to prove address
ownership by issuing a secured NUD_NSOL, which has to be answered
with a secured NUD_NADV by the probed node.

SEND SAVI devices issue secured NUD_NSOL messages periodically in
order to refresh bindings, which have to be answered with a valid
NUD_NADV message by the node for which the binding exists.

SEND SAVI devices only forward packets with off-link source addresses
if they are received from a port manually configured to connect to a
router.

SEND SAVI needs to be protected against replay attacks, i.e., attacks
in which a secured SEND message is replayed by another node. As
discussed before, the SEND SAVI specification uses SEND messages to
create a binding between the address contained in the message (that
must be signed by a node possessing the private key associated to the
address) and the port through which the message is received. If an
attacker manages to obtain such a message from another node, for
example, because the message was sent to the all-nodes multicast
address or because the attacker has subscribed to the Solicited Node
multicast address associated to a remote node, it could replay it
preserving the original signature. This may create an illegitimate
binding in the SEND SAVI device or could be used to abort address
configuration at the other node. While SEND provides some means to
limit the impact of the replay of ND messages, the emphasis for SEND
anti-replay protection is to limit to a short period of time the
validity of the ND information transmitted in the message, for
example, the relationship between an IPv6 address and a layer-2
address. Note that the period must be long enough to assure that the
information sent by the legitimate sender is considered valid despite
the possible differences in clock synchronization between the sender
and receiver(s). For example, with the values recommended by
[RFC3971] for TIMESTAMP_FUZZ and TIMESTAMP_DRIFT, a node receiving a
DAD_NSOL message would not discard replays of this message being
received within a period of approximately 2 seconds (more precisely,
2/0.99 seconds). The underlying assumption for SEND security is that
even if the message is replayed by another node during this period of
time, the information disseminated by ND is still the same. However,
allowing a node to replay a SEND message does have an impact on the
SEND SAVI operation, regardless of the time elapsed since it was
generated, since the node can create a new binding in a SEND SAVI
device for the port to which an illegitimate node attaches. As can
be concluded, the protection provided by SEND is not enough in all
cases for SEND SAVI.

SEND SAVI increases the protection against the replay attacks
compared to SEND. First, each node is required to connect to the
SEND SAVI topology through a different port to prevent eavesdropping
before entering the SAVI protection perimeter. Then, SEND SAVI
bindings are updated only according to messages whose dissemination
can be restricted in the SEND SAVI topology without interfering with
the normal SEND operation. The messages used by SEND SAVI to create
bindings are DAD_NSOL messages, for which SEND SAVI limits its
propagation to the ports through which a previous binding for the
same IPv6 address existed (see Section 3.3.2), and NUD_NADV messages

in response to a secured NUD_NSOL sent by the SEND SAVI device only
through the tested port. Finally, SEND SAVI filtering rules prevent
nodes from replaying messages generated by the SEND SAVI devices
themselves. Section 5.1 discusses in more detail the protection
provided by SEND SAVI against replay attacks.

2.3. SEND SAVI Protection Perimeter

In order to reduce computing and state requirements in SEND SAVI
devices, SEND SAVI devices can be deployed to form a "protection
perimeter" [RFC7039]. With this deployment strategy, SEND SAVI
devices perform source-address validation only when packets enter in
the protected realm defined through the protection perimeter. The
perimeter is defined by appropriate configuration of the roles of
each port, which can be 'Validating' or 'Trusted':

o Validating ports (VPs) are ports in which SEND SAVI filtering and
   binding creation are performed.

o Trusted ports (TPs) are ports in which limited processing is
   performed. Only SEND messages related with certificates, prefix
   information, and DAD operation are processed in order to update
   the state of the SEND SAVI device or the state related with any of
   the Validating ports of the switch.

Figure 1 shows a typical topology involving trusted and untrusted
infrastructure.

```
     +--+    +--+                        +--+    +--+
     |H1|    |H2|                        |H3|    |R1|
     +--+    +--+                        +--+    +--+
      |       |                           |       |
   +-----------SEND SAVI PROTECTION PERIMETER-----------+
   |  |       |                   |       |       |     |
   |  +-1-----2-+                 |     +-1-----2-+     |
   |  |  SEND-  |                 |     |  SEND-  |      |
   |  |  SAVI1  |                 |     |  SAVI2  |      |
   |  +-3--4----+                 |     +--3--4---+      |
   |    |  |      +--------------+      |    |           |
   |    |  +------+-----------   |      |    |           |
   |    |         |   SWITCH-A   +--------+  |           |
   |    |  +------+-----------   |      |    |           |
   |    |  |      +--------------+      |    |           |
   |  +-1--2----+                       |  +-----1---+   |
   |  |  SEND-  |                       |  |  SEND-  |   |
   |  |  SAVI3  |                       |  |  SAVI4  |   |
   |  +-3-----4-+                       |  +----4----+   |
   |    |     |                         |       |        |
   +-----------SEND SAVI PROTECTION PERIMETER-----------+
        |     |                                 |
      +--+    +--+                            +--+
      |R2|    |H4|                            |H5|
      +--+    +--+                            +--+
```

                  Figure 1: SAVI Protection Perimeter

   Trusted ports are used for connections with trusted infrastructures,
   such as routers and other SEND SAVI devices. Port 2 of SEND-SAVI2
   and port 3 of SEND-SAVI3 are Validating ports because they connect to
   routers. Port 3 of SEND-SAVI1 and port 1 of SEND-SAVI3 as well as
   port 4 of SEND-SAVI2 and port 1 of SEND-SAVI4 are trusted because
   they connect two SAVI devices. Finally, port 4 of SEND-SAVI1, port 3
   of SEND-SAVI2, and port 2 of SEND-SAVI3 are trusted because they
   connect to SWITCH-A to which only trusted nodes are connected.

   Validating ports are used for connection with non-trusted
   infrastructures; therefore, hosts connect normally to Validating
   ports. So, in Figure 1 above, ports 1 and 2 of SEND-SAVI1, port 1 of
   SEND-SAVI2, and port 4 of SEND-SAVI3 are Validating ports because
   they connect to hosts. Port 4 of SEND-SAVI4 is also a Validating
   port because it is connected to host H5.

For a more detailed discussion on this, see Section 3.4.

2.4. Special Cases

Multi-subnet links: In some cases, a given subnet may have several
   prefixes. This is supported by SEND SAVI as any port can support
   multiple prefixes.

Multihomed hosts: A multihomed host is a host with multiple
   interfaces. The interaction between SEND SAVI and multihomed
   hosts is as follows. If the different interfaces of the host are
   assigned different IP addresses and packets sent from each
   interface and always carry the address assigned to that interface
   as the source address, then from the perspective of a SEND SAVI
   device, this is equivalent to two hosts with a single interface,
   each with an IP address. SEND SAVI supports this without
   additional considerations. If the different interfaces share the
   same IP address or if the interfaces have different addresses but
   the host sends packets using the address of one of the interfaces
   through any of the interfaces, then SEND SAVI does not directly
   support it. It would require either connecting at least one
   interface of the multihomed host to a Trusted port or manually
   configuring the SEND SAVI bindings to allow binding the address of
   the multihomed host to multiple anchors simultaneously.

Virtual switches: A hypervisor or a host operating system may
   perform bridging functions between virtual hosts running on the
   same machine. The hypervisor or host OS may in turn connect to a
   SEND SAVI system. This scenario is depicted in Figure 2, with two
   virtual machines, VM1 and VM2, connected through a virtual switch,
   VS1, to SEND SAVI device SEND-SAVI1. The attachment points of VS1
   to VM1 and VM2 are configured as Validating.

```
Host1
+-----------------+
| +---+   +---+   |
| |VM1|   |VM2|   |
| +---+   +---+   |
|   |       |     |
| +-1-----2--+    |
| |    VS1   |    |
| +--3-------+    |
|    |            |
+----|-----------+
     |
     |
   +--1-----2--+
   |   SEND-   |
   |   SAVI1   |
   +--3---4----+
      |   |
```
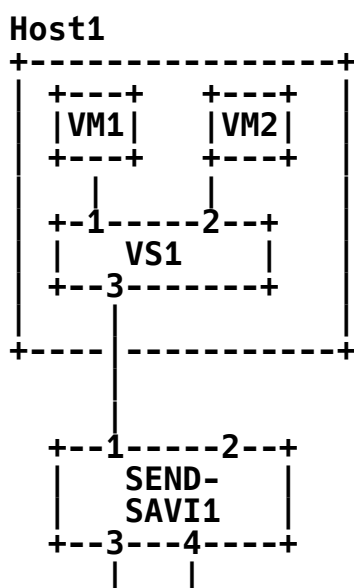
        Figure 2: Virtual Switches Connected to the SEND SAVI Device

   In order to provide proper security against replay attacks,
   performing SEND SAVI filtering as close to untrusted hosts as
   possible (see Sections 3.4 and 5.1) is recommended. In this
   scenario, this objective can be achieved by enabling SEND SAVI
   validation in VS1. Ideally, VS1 could be integrated into the SEND
   SAVI protection perimeter if the hypervisor or host OS at Host1 can
   be trusted (even though VM1 and VM2 could not be trusted). To do so,
   both the attachment to SEND-SAVI1 at VS1, and port 1 at SEND-SAVI1,
   are configured as Trusted.

   If the administrator of the network does not trust VS1, port 1 of
   SEND-SAVI1 is configured as Validating, so that every address being
   used at Host1 is validated at SEND-SAVI1 by SEND SAVI. The
   attachment point to the physical network at VS1 should be configured
   as Trusted if the host administrator knows that it is connected to a
   SEND SAVI device; in this case, VS1 relies on the infrastructure
   comprised by the physical SEND SAVI devices but not vice versa.
   Packets egressing from VM1 are validated twice: first at VS1 and then
   at SEND-SAVI1. Packets going in the reverse direction (from an
   external host to VM1) are validated once: when they first reach a
   SEND SAVI device. If the administrator of VS1 does not trust the
   physical switch to which it attaches, it can configure the attachment
   to SEND-SAVI1 as Validating. In Figure 2 above, this means that a
   packet going from another host to VM1 would be validated twice: once
   when entering the SEND SAVI perimeter formed by the physical devices
   and again when entering at VS1.

Untrusted routers: One can envision scenarios where routers are
dynamically attached to a SEND SAVI network. A typical example would
be a mobile phone connecting to a SEND SAVI switch where the mobile
phone is acting as a router for other personal devices that are
accessing the network through it. Regarding the validation of the
source address performed in a SEND SAVI device, such an untrusted
router does not seem to directly fall in the category of trusted
infrastructure (if this was the case, it is likely that all devices
would be trusted); hence, it cannot be connected to a Trusted port,
and if it is connected to a Validating port, the SEND SAVI switch
would discard all the packets containing an off-link source address
coming from that device. Although the SEND SAVI device to which this
router attaches could be configured to permit the transit of packets
with source addresses belonging to the set of prefixes reachable
through the untrusted router, such a mechanism is out of the scope of
this document. As a result, the default mechanism described in this
specification cannot be applied in such a scenario.

3. SEND SAVI Specification

3.1. SEND SAVI Data Structures

The following three data structures are defined for SEND SAVI
operations.

SEND SAVI Database: The SEND SAVI function relies on state
information binding the source IPv6 address used in data packets to
the port through which the legitimate node connects. Such
information is stored in the SEND SAVI Database. The SEND SAVI
Database is populated with the contents of validated SEND messages.
Each entry contains the following information:

o IPv6 source address

o Binding anchor: the port through which the packet was received

o Lifetime

o Status: TENTATIVE_DAD, TENTATIVE_NUD, VALID, TESTING_VP,
   TESTING_VP'

o Alternative binding anchor: the port from which a DAD_NSOL message
   or any data packet has been received while a different port was
   stored in the binding anchor for the address.

o Creation time: the value of the local clock when the entry was
   first created

SEND SAVI Prefix List: SEND SAVI devices need to know which ones are
the link prefixes in order to identify local and off-link traffic. A
SEND SAVI device MUST support discovering this information from the
Prefix Information option [RFC4861] with the L bit set of Router
Advertisement (RADV) messages coming from Trusted ports, as described
in Section 3.3.2. The list of prefixes MAY also be configured
manually. This information is not specific to a given port. The
SEND SAVI Prefix List contains one entry per prefix in use, as
follows:

o Prefix: the prefix included in a Prefix Information option.

o Prefix lifetime: time in seconds that the prefix is valid.
   Initially set to the Valid Lifetime value of the Prefix
   Information option of a valid RADV message or set to a value of
   all 1 bits (0xffffffff), which represents infinity, if configured
   manually.

When the SEND SAVI device boots, it MUST send a Router Solicitation
(RSOL) message, which does not need to be secured if the unspecified
address is used (see [RFC3971], Sections 5.1.1 and 5.2.1). The SAVI
device SHOULD issue a RSOL message in case the prefix entry is about
to expire.

3.2. SEND SAVI Device Configuration

In order to perform the SEND SAVI operation, some basic parameters of
the SEND SAVI device have to be configured. Since a SEND SAVI device
operates as a SEND node to generate NUD_NSOL, RSOL, or Certification
Path Solicitation (CPS) messages:

o The SEND SAVI device MUST be configured with a valid CGA address.
   When the SEND SAVI device configures this address, it MUST behave
   as a regular SEND node, i.e., using secured NSOL messages to
   perform DAD, etc., in addition to fulfilling the requirements
   stated for regular IPv6 nodes [RFC6434].

o The SEND SAVI device MAY be configured with at least one trust
   anchor if it is configured to validate RADV messages (see
   Section 3.3.2). In this case, the SEND SAVI device MAY be
   configured with certification paths. The alternative is obtaining
   them by means of issuing Certification Path Solicitation messages,
   as detailed in the SEND specification [RFC3971].

In addition, the port role for each port of the SEND SAVI device MUST
be configured. The guidelines for this configuration are specified
in Section 3.4.

3.3. Traffic Processing

In this section, we describe how packets are processed by a SEND SAVI
device. Behavior varies depending on if the packet belongs to local
or transit traffic. This is determined by checking if the prefix of
the source address is included in the SEND SAVI Prefix List or in the
unspecified address (local traffic) or not included in the SEND SAVI
Prefix List (transit traffic).

3.3.1. Transit Traffic Processing

Transit traffic processing occurs as follows:

o If the SEND SAVI device receives a transit traffic packet through
   a Trusted port, it forwards it without any SAVI processing.

o If the SEND SAVI device receives a transit traffic packet through
   a Validating port, it discards the packet.

3.3.2. Local Traffic Processing

If the verification of the source address of a packet shows that it
belongs to local traffic, this packet is processed using the state
machine described in this section.

For the rest of the section, the following assumptions hold:

o When it is stated that a secured NUD_NSOL message is issued by a
   SEND SAVI device through a port P, it means that the SEND SAVI
   device generates a NUD_NSOL message, according to the Neighbor
   Unreachability Detection procedure described in [RFC4861],
   addressed to the IPv6 target address, which is the source address
   of the packet triggering the procedure. This message is secured
   by SEND as defined in [RFC3971]. The source address used for
   issuing the NUD_NSOL message is the source address of the SEND
   SAVI device. The message is sent only through port P.

o When it is stated that a validated NUD_NADV message is received by
   a SEND SAVI device, it means that a SEND secured NUD_NADV message
   has been received by the same port P through which the
   corresponding NUD_NSOL message was issued, and the NUD_NADV
   message has been validated according to [RFC3971] to prove
   ownership for the IPv6 address under consideration and to prove
   that it is a response for the previous NUD_NSOL message issued by
   the SEND SAVI device (containing the same nonce value as the
   NUD_NSOL message to which it answers).

   We use VP to refer to a Validating port and TP to refer to a Trusted
   port.

   The state machine is defined for a binding of a given source IPv6
   address in a given SEND SAVI device. In the transitions considered,
   packets described as inputs refer to the IPaddr IPv6 address
   associated to the state machine.

   The possible states for a given IPaddr are NO_BIND, TENTATIVE_DAD,
   TENTATIVE_NUD, VALID, TESTING_VP, and TESTING_VP'. The NO_BIND state
   represents that no binding exists for IPaddr; this is the state for
   all addresses unless a binding is explicitly created.

   The states can be classified into 'forwarding' states, i.e., states
   in which packets received from the port associated to the IPv6
   address are forwarded, and 'non-forwarding' states, i.e., states in
   which packets different to the ones used for signaling are not
   forwarded. VALID, TENTATIVE_DAD, TESTING_VP, and TESTING_VP' are
   forwarding states, and NO_BIND and TENTATIVE_NUD are non-forwarding
   states.

   The SEND SAVI device MUST join the Solicited Node Multicast group for
   all the addresses whose state is other than NO_BIND. This is needed
   to make sure that the SEND SAVI device receives DAD_NSOL messages
   issued for those addresses. Note that it may not be enough to relay
   on the Multicast Listener Discovery (MLD) messages being sent by the
   node attached to a Validating port for which a binding for the
   corresponding address exists, since the node may move and packets
   sent to that particular Solicited Node Multicast group may no longer
   be forwarded to the SEND SAVI device.

   In order to determine which traffic is on-link and off-link, the SEND
   SAVI device MUST support discovery of this information from the
   Prefix Information option with the L bit set of RADV messages. In
   this case, at least one router SHOULD be configured to advertise RADV
   messages containing a Prefix Information option with the prefixes
   that the untrusted nodes can use as source addresses, and the bit L
   set. An alternative to this is to manually configure the SEND SAVI
   Prefix List or restrict the use of link-local addresses.

   SEND SAVI devices MUST discard RADV messages received from Validating
   ports. RADV messages are only accepted and processed when received
   through Trusted ports.

   SEND SAVI devices SHOULD NOT validate RADV messages to update the
   SEND SAVI Prefix List and forward them to other nodes. These
   messages can only be received from Trusted ports, and we assume that
   routers are trusted. Validating RADV messages would be required in

any SEND SAVI device the node is traversing. Besides, hosts will
validate this message before using the information it contains.

In case SEND SAVI devices are configured to validate RADV messages,
SEND SAVI devices SHOULD support the processing of validated
Certification Path Advertisement (CPA) messages, sent in reply to CPS
messages, to acquire certificates used to validate router messages;
alternatively, it SHOULD be configured with a certification path.

The state machine defined for the SEND SAVI operation adheres to the
following design guidelines:

o The only events that trigger state changes from forwarding to non-
   forwarding states, and vice versa, are the reception of DAD_NSOL,
   DAD_NADV, and NUD_NADV or the expiration of a timer. The other
   possible input to consider is 'any other packet', which could
   generate changes to states belonging to the same forwarding or
   non-forwarding class as the original state. In other words, when
   'any other packet' is received, the state cannot move from
   forwarding to non-forwarding, and vice versa. The reduced set of
   messages being able to trigger a change simplifies the processing
   at SEND SAVI devices.

o DAD_NADV and NUD_NADV are only processed when they are a response
   to a DAD_NSOL or a NUD_NSOL message.

o SEND SAVI devices MUST only use ND messages received through
   Validating ports if they are valid; otherwise, they discard them.
   SEND SAVI devices SHOULD assume that such messages received from
   Trusted ports have been validated by other SEND SAVI devices, or
   come from a trusted device such a router, so they SHOULD NOT
   attempt to validate them in order to reduce the processing load at
   the SEND SAVI device.

o The only messages the SEND SAVI device is required to generate
   specifically per each source IP address are MLD and NUD_NSOL
   messages. This also keeps the state machine simple.

o Well-behaved nodes are expected to initiate communication by
   sending secured DAD_NSOL messages. The SEND SAVI state machine is
   tailored to efficiently process these events. The reception of
   other packet types without receiving previously validated DAD_NSOL
   messages is assumed to be a consequence of bad-behaving nodes or
   infrequent events (such as packet loss, a change in the topology
   connecting the switches, etc.). While a binding will ultimately
   be created for nodes affected by such events, simplicity of the
   state machine is prioritized over any possible optimization for
   these cases.

   o If a node has a configured address, and it can prove that it owns
      this address, the binding is preserved regardless of any
      indication that a binding for the same source address could be
      configured in other SEND SAVI devices. Bindings for the same
      source address in two or more SEND SAVI devices may occur due to
      several reasons, for example, when a host moves (the two bindings
      exist just for a short period of time) or when many nodes generate
      the same address and the DAD procedure has failed. In these
      infrequent cases, SEND SAVI preserves connectivity for the
      resulting bindings.

   Next, we describe how different inputs are processed, depending on
   the state of the binding of the IP address 'IPaddr'. Note that every
   ND message is assumed to be validated according to the SEND
   specification.

   To facilitate the reader's understanding of the most relevant
   transitions of the SEND SAVI state machine, a simplified version,
   which does not contain every possible transition, is depicted in
   Figure 3:

```
                        +-------------+
                        |             |
                        | TESTING_VP' |
                        |             |
                        +-------------+
        Timeout/VP=VP'    |         ^
                          |         |
        VP_NUD_NADV/-     |         |    VP'_DAD_NSOL/
                          |         |    VP_NUD_NSOL
                          |         |
                          v         |
   VP_DAD_NSOL/-        +---------+
        +-------------  |         |
        |               |  VALID  |< -------------------+
        |    +--------- >|         |                    |
        |    |           +---------+                    |
        |    |            ^       |                     |
        |    |  VP_NUD_    |       |  Timeout,           |
        |    |  NADV/-     |       |  TP_DAD_NSOL/VP_NUD_NSOL
        |    |            |       v                     |
        |    |           +-------------+                |
        |    |           |             |                |
        |    |           | TESTING_VP  |                |
        |    |           |             |                |
        |    |           +-------------+                |
        |    |                  |                        |
        |    |  VP*,            | Timeout/-              |
        |    |  Timeout/-       |            VP_NUD_NADV/-
        |    |                  |                        |
        v    |                  |       +----------------+
   +---------------+            |       |                |
   |               |            |       | TENTATIVE_NUD  |
   | TENTATIVE_DAD |            |       |                |
   |               |            |       +----------------+
   +---------------+            |        |             ^
        ^    |                  |        |             |
        |    |  TP_DAD_NSOL,    | Timeout/-           |
        |    |  TP_DAD_NADV/-   |                      |
        |    |                  v                      |
        |    +--------- >+---------+                   |
        |               |         |< -----+            |
        |               | NO_BIND |       |            |
        +---------------|         |-------+------------+
   VP_DAD_NSOL/-        +---------+        VP*/VP_NUD_NSOL
```
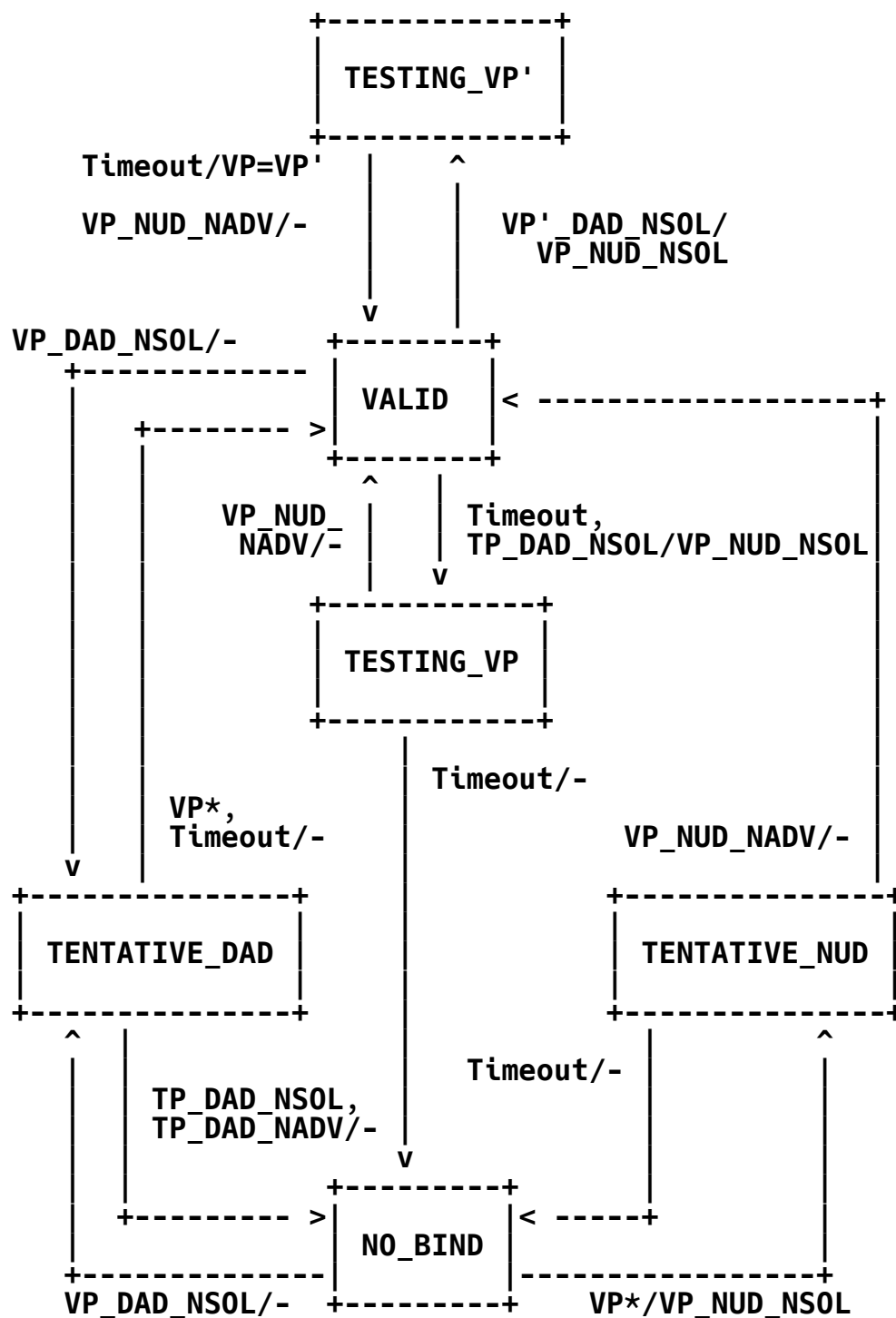
              Figure 3: Simplified SEND SAVI State Machine

Each state transition is characterized by any of the events that may
trigger the change and the message(s) generated as a result of this
change. The meaning of some terms are referred next:

o VP_DAD_NSOL as a triggering event means that a validated DAD_NSOL
   message has been received from the current BINDING_ANCHOR port VP.

o VP* means any packet (data packet) received from the current
   BINDING_ANCHOR port VP.

o TP_DAD_NSOL as a triggering event means that a DAD_NSOL message
   was received from a Trusted port.

o - means that no message is sent. VP=VP' means that the
   BINDING_ANCHOR is set to VP'.

The notation

    Timeout, TP_DAD_NSOL/VP_NUD_NSOL

means that the transition is triggered by either a timeout expiration
or the reception of a DAD_NSOL message from a Trusted port, and in
addition to the transition, a NUD_NSOL message is sent through port
VP.

For the rest of the description, we assume the following:

o When a validated message is required (i.e., a 'validated
   DAD_NSOL'), messages are check for validity in the considered
   switch according to [RFC3971], and messages not fulfilling these
   conditions are discarded.

o When any SEND message is received from a validated port, the SEND
   SAVI SHOULD assume that the message has been validated by the SEND
   SAVI device through which the message accessed the SEND SAVI
   protection perimeter (unless the SEND SAVI perimeter has been
   breached), or the device generating it is trusted. In this case,
   the SAVI device does not perform any further validation.
   Performing validation for SEND messages received through a Trusted
   port may affect performance negatively.

NO_BIND

When the node is in this state, there are no unresolved NUD_NSOL
messages generated by SEND SAVI or DAD_NSOL propagated to any
Validating port, so the only relevant inputs are DAD_NSOL messages
coming either from a Validating port (VP) or Trusted port (TP), or
any packet other than DAD_NSOL coming from a VP or TP. There are no
timers configured for this state.

Messages received from a Validating port:

o If a validated DAD_NSOL message is received from a Validating port
   VP, the SEND SAVI device forwards this message to all appropriate
   Trusted ports (the subset of Trusted ports that belong to the
   forwarding layer-2 topology, with the restrictions imposed by the
   MLD snooping mechanism, if applied). DAD_NSOL messages are not
   sent through any of the ports configured as Validating ports. The
   SEND SAVI device sets the LIFETIME to TENT_LT, stores all the
   information required for future validation of the corresponding
   DAD_NADV message (such as the nonce of the message), creates a new
   entry in the SEND SAVI Database for IPaddr, sets BINDING_ANCHOR to
   VP, and changes the state to TENTATIVE_DAD. Creation time is set
   to the current value of the local clock.

   Note that in this case, it is not possible to check address
   ownership by sending a NUD_NSOL because while the node is waiting
   for a possible DAD_NADV, its address is in tentative state and the
   node cannot respond to NSOL messages [RFC4862].

o If any packet other than a DAD_NSOL is received through a
   Validating port VP, the SEND SAVI device issues a secured NUD_NSOL
   through port VP. The SEND SAVI device sets the LIFETIME to
   TENT_LT. The SEND SAVI device creates a new entry in the SEND
   SAVI Database for IPaddr, sets BINDING_ANCHOR to VP, and the state
   is changed to TENTATIVE_NUD. Creation time is set to the current
   value of the local clock. The SAVI device MAY discard the packet
   while the NUD procedure is being executed or MAY store it in order
   to send it if the next transitions are (strictly) TENTATIVE_NUD
   and then VALID.

Messages received from a Trusted port:

o If a DAD_NSOL message containing IPaddr as the target address is
   received through a Trusted port, it MUST NOT be forwarded through
   any of the Validating ports: it is sent through the proper Trusted
   ports. The state is not changed.

   o Any packet other than a DAD_NSOL received from a Trusted port is
      forwarded to its destination. This packet is assumed to come from
      a SEND SAVI device that has securely validated the binding,
      according to the SEND SAVI rules (unless the SEND SAVI perimeter
      has been breached). The state is not changed.

   TENTATIVE_DAD

   To arrive at this state, the SEND SAVI device has received a
   validated DAD_NSOL coming from the BINDING_ANCHOR port, and it has
   forwarded it to the appropriate TPs. The relevant events occurring
   in this state are the reception of a DAD_NADV message from a TP, a
   DAD_NSOL message from the BINDING_ANCHOR port, other Validating port
   or TP, a data packet from the BINDING_ANCHOR port, and the expiration
   of the LIFETIME timer initiated when the DAD_NSOL was received at the
   BINDING_ANCHOR port.

   Messages received from a Trusted port:

   o The reception of a valid DAD_NADV message from a Trusted port
      indicates that the binding cannot be configured for the
      BINDING_ANCHOR port. The state is changed to NO_BIND, and the
      LIFETIME is cleared.

   o The reception of a valid DAD_NSOL from a Trusted port indicates
      that a node connected to another SEND SAVI device may be trying to
      configure the same address at the same time. The DAD_NSOL message
      is forwarded to the BINDING_ANCHOR port, so that the node at this
      port will not configure the address, as stated in [RFC4862]. The
      DAD_NSOL message is also forwarded to all appropriate Trusted
      ports. Then, the LIFETIME is cleared, and the state is changed to
      NO_BIND.

   o Any packet other than a validated DAD_NSOL or DAD_NADV received
      from a Trusted port is forwarded to its destination. This packet
      is assumed to come from a SEND SAVI device that has securely
      validated the binding, according to the SEND SAVI rules (unless
      the SEND SAVI perimeter has been breached). The state is not
      changed.

Messages received from a Validating port different from the
BINDING_ANCHOR:

o A validated DAD_NSOL is received from a Validating port VP'
   different from the BINDING_ANCHOR port. The reception of a valid
   DAD_NSOL from port VP' indicates that a node connected to VP' may
   be trying to configure the same address at the same time. The
   DAD_NSOL message is forwarded to the BINDING_ANCHOR port, so that
   the node at this port will not configure the address, as stated in
   [RFC4862]. The DAD_NSOL message is also forwarded to all
   appropriate Trusted ports. Then, the BINDING_ANCHOR is set to VP'
   (through which the DAD_NSOL message was received), the LIFETIME is
   set to TENT_LT, and the state remains in TENTATIVE_DAD.

o Any packet other than a validated DAD_NSOL received from a
   Validating port VP' different from the BINDING_ANCHOR port is
   discarded. The state is not changed.

Messages received from the BINDING_ANCHOR port:

o If a validated DAD_NSOL is received from the BINDING_ANCHOR port,
   the LIFETIME is set to TENT_LT, and the state remains in
   TENTATIVE_DAD.

o If any packet other than a DAD_NSOL is received from the
   BINDING_ANCHOR port, it is assumed that the node has configured
   its address, although it has done it in less time than expected by
   the SEND SAVI device (less than TENT_LT). Since the node proved
   address ownership by means of the validated DAD_NSOL message, the
   LIFETIME is set to DEFAULT_LT, and the state is changed to VALID.

LIFETIME expires:

o If LIFETIME expires, it is assumed that no other node has
   configured this address. Therefore, the Validating port VP
   (currently stored in the BINDING_ANCHOR) could be bound to this
   IPv6 address. The LIFETIME is set to DEFAULT_LT, and the state is
   changed to VALID.

VALID

To arrive at this state, the SEND SAVI device has successfully
validated address ownership and has created a binding for IPaddr.
Relevant transitions for this state are triggered by the reception of
DAD_NSOL from the BINDING_ANCHOR port, other Validating port or a TP,
and any packet other than DAD_NSOL from a Validating port other than

the BINDING_ANCHOR or a TP. The expiration of LIFETIME is also
relevant to trigger a check for address ownership for the node at the
BINDING_ANCHOR port.

Messages received from the BINDING_ANCHOR port:

o If a validated DAD_NSOL with IPaddr as a source address is
   received through the BINDING_ANCHOR port, it is forwarded to the
   appropriate Trusted ports. The LIFETIME is set to TENT_LT, and
   the state is changed to TENTATIVE_DAD.

o Any packet other than a DAD_NSOL containing IPaddr as a source
   address arriving from the BINDING_ANCHOR port is forwarded
   appropriately. The state is not changed.

Messages received from a Trusted port:

o If a DAD_NSOL with IPaddr as a source address is received through
   a Trusted port, the message is forwarded to VP. The LIFETIME is
   set to TENT_LT, a secured NUD_NSOL message is sent to IPaddr
   through VP, and the state is changed to TESTING_VP.

o If any packet other than a DAD_NSOL with IPaddr as a source
   address is received through a Trusted port, the packet is
   forwarded to VP and to other appropriate Trusted ports. A secured
   NUD_NSOL is sent to the BINDING_ANCHOR port, the LIFETIME is set
   to TENT_LT, and the state is changed to TESTING_VP.

Messages received from a Validating port different from the
BINDING_ANCHOR:

o If a validated DAD_NSOL packet with IPaddr as a source address is
   received through a Validating port VP' (a VP' different from the
   current BINDING ANCHOR), the message is forwarded to the
   BINDING_ANCHOR port. In addition, a secured NUD_NSOL is sent to
   the BINDING_ANCHOR port, the ALTERNATIVE BINDING ANCHOR is set to
   port VP' (for future use if the node at VP' is finally selected),
   the LIFETIME is set to TENT_LT, and the state is changed to
   TESTING_VP'.

o If any packet other than a DAD_NSOL with IPaddr as a source
   address is received from a Validating port VP', different from the
   current BINDING_ANCHOR for this binding, VP, the packet is
   discarded. The SEND SAVI device MAY issue a secured NUD_NSOL
   through the BINDING_ANCHOR port, store VP' in the ALTERNATIVE
   BINDING ANCHOR for possible future use, set the LIFETIME to
   TENT_LT, and change the state to TESTING_VP'. An alternative to
   this behavior is that the SEND SAVI device MAY not do anything (in

this case, the state would eventually change after a maximum
DEFAULT_LT time; if the node at VP does not respond to a NUD_NSOL
at TESTING_VP, the state is moved to NO_BIND). Then, a packet
arriving from VP' would trigger a process that may end up with
binding for the node connecting to VP'.

LIFETIME expires:

o If LIFETIME expires, a secured NUD_NSOL message is sent through
   the BINDING_ANCHOR port to IPaddr, the LIFETIME is set to TENT_LT,
   and the state is changed to TESTING_VP. In the TESTING_VP state,
   packets are still being forwarded until the timer expires without
   receiving a NUD_NADV.

TESTING_VP

When the SEND SAVI device enters the TESTING_VP state, the current
Validating port is under check through a secured NUD_NSOL message
generated by the SEND SAVI device. While testing, packets from the
current Validating port are forwarded. Packets coming from Trusted
ports are also forwarded. The relevant events for this state are the
reception of a NUD_NADV message from VP; the reception of a DAD_NSOL
message from VP, VP', or TP; the reception of any packet other than
the previous cases from VP, VP', or TP; and the expiration of the
timer associated to the reception of NUD_NADV.

Messages received from the BINDING_ANCHOR port:

o If a validated NUD_NADV is received from VP, the LIFETIME is
   changed to DEFAULT_LT, and the state is changed to VALID. The
   message is not forwarded to any other port.

o If a validated DAD_NSOL message is received from VP, it is
   forwarded to the appropriate Trusted ports, the LIFETIME is set to
   DEFAULT_LT, and the state is changed to TENTATIVE_DAD.

o Any packet other than DAD_NSOL or NUD_NADV containing IPaddr as a
   source address arriving from the BINDING_ANCHOR port is forwarded.
   Neither the LIFETIME nor the state are changed.

Messages received from a Trusted port:

o If a DAD_NSOL packet is received from a Trusted port, the message
   is forwarded to VP and the appropriate Trusted ports. Neither the
   LIFETIME nor the state are changed. The node at the
   BINDING_ANCHOR port is under check; if it still is at this port,
   it should answer with a NUD_NADV and also with a DAD_NADV. If it
   is not there, neither the NUD_NADV nor the DAD_NADV will be

received, the timer will expire, and the local state will move to
NO_BIND.

o If a packet other than a DAD_NSOL arrives from a Trusted port, the
   packet is forwarded. Neither the LIFETIME nor the state are
   changed.

Messages received from a Validating port different from the
BINDING_ANCHOR:

o If a valid DAD_NSOL is received from a Validating port VP' other
   than the current BINDING_ANCHOR port, the message is forwarded to
   the BINDING_ANCHOR port and to the appropriate Trusted ports. In
   addition, a secured NUD_NSOL is sent to the BINDING_ANCHOR port,
   the ALTERNATIVE BINDING_ANCHOR is set to VP' (for future use if
   the node at VP' is finally selected), the LIFETIME is set to
   TENT_LT, and the state is changed to TESTING_VP'.

o Any other packet received from a Validating port VP' other than
   the BINDING_ANCHOR port is discarded. This may occur because the
   node has moved but has not issued a DAD_NSOL or the DAD_NSOL
   message has been lost. The state will eventually move to NO_BIND,
   and then the packets sent from VP' will trigger the creation of
   the binding for VP'.

LIFETIME expires:

o If the LIFETIME expires, the LIFETIME is cleared and the state is
   changed to NO_BIND.

TESTING_VP'

To arrive at this state, the SEND SAVI device has received an
indication that a node at VP' different from the BINDING_ANCHOR port
wants to send data with IPaddr as a source address and has occurred
while a binding existed for VP. The port VP' that triggered the
change of the state to TESTING_VP' was stored at the
ALTERNATIVE_BINDING_ANCHOR, so that it can be retrieved if the node
at VP' is determined as the legitimate owner of IPaddr. The SEND
SAVI device has issued a NUD_NSOL to IPaddr through the
BINDING_ANCHOR port. The relevant events that may occur in this case
are the reception of a NUD_NADV from port VP (the BINDING_ANCHOR
port); the reception of a DAD_NSOL from VP, VP', TP, and VP" (VP"
different from VP and VP'); the reception of any other packet from
VP, VP', TP, or VP"; and the expiration of the timer.

Messages received from the BINDING_ANCHOR port:

o A validated NUD_NADV is received from the BINDING_ANCHOR port.
   The reception of a valid NUD_NADV indicates that the node at VP is
   defending its address. The BINDING_ANCHOR in use is kept, the
   LIFETIME is set to DEFAULT_LT, and the state is changed to VALID.

o If a valid DAD_NSOL is received from the BINDING_ANCHOR port, it
   is forwarded to VP' (the port stored in the
   ALTERNATIVE_BINDING_ANCHOR). The BINDING_ANCHOR in use is kept,
   the LIFETIME is set to TENT_LT, and the state is changed to
   TENTATIVE_DAD. When the DAD_NSOL message is received by the node
   at VP', the address will not be configured.

o Any packet other than a validated DAD_NSOL, or a validated
   NUD_NADV coming from the BINDING_ANCHOR port, is forwarded, and
   the state is not changed.

Messages received from the ALTERNATIVE_BINDING_ANCHOR Validating
port:

o If a valid DAD_NSOL is received from the port stored in the
   ALTERNATIVE_BINDING_ANCHOR, it is forwarded to the BINDING_ANCHOR
   port. The BINDING_ANCHOR and the ALTERNATIVE BINDING ANCHOR are
   kept, the LIFETIME is set to DEFAULT_LT, and the state is not
   changed.

o Any packet other than a validated DAD_NSOL coming from the
   ALTERNATIVE_BINDING_ANCHOR port is discarded, and the state is not
   changed.

Messages received from a Validating port different from the
BINDING_ANCHOR and the ALTERNATIVE_BINDING_ANCHOR ports:

o If a validated DAD_NSOL is received from port VP", different from
   BINDING_ANCHOR and the ALTERNATIVE_BINDING_ANCHOR ports, it is
   forwarded to the BINDING_ANCHOR and the ALTERNATIVE_BINDING_ANCHOR
   ports. The node at the ALTERNATIVE BINDING ANCHOR port is
   expected to unconfigure its address if the message triggering the
   transition to this state was a DAD_NSOL message received from the
   ALTERNATIVE_BINDING_ANCHOR port (and not any other packet). The
   state remains in TESTING_VP', although VP" is stored in the
   ALTERNATIVE_BINDING_ANCHOR for future use if the node at VP" is
   finally selected. The LIFETIME is not changed.

o Any packet other than a validated DAD_NSOL received from port VP"
   is discarded and does not affect the state.

Messages received from a Trusted port:

o If a DAD_NSOL is received from a Trusted port, the message is
   forwarded to the BINDING_ANCHOR, ALTERNATIVE_BINDING_ANCHOR ports,
   and other appropriate Trusted ports. The LIFETIME is left
   unchanged, and the state is changed to TESTING_VP. The node at
   the ALTERNATIVE_BINDING_ANCHOR port is expected to unconfigure its
   address if the packet triggering the transition to this state was
   a DAD_NSOL message received from the ALTERNATIVE_BINDING_ANCHOR
   port.

o Any packet other than a DAD_NSOL coming from a Trusted port is
   forwarded appropriately, but the state is not changed.

LIFETIME expires:

o If LIFETIME expires, it is assumed that the node for which the
   binding existed is no longer connected through the BINDING_ANCHOR
   port. Therefore, the BINDING_ANCHOR is set to the
   ALTERNATIVE_BINDING_ANCHOR port value. The LIFETIME is set to
   DEFAULT_LT, and the state is changed to VALID.

TENTATIVE_NUD

To arrive at this state, a data packet has been received through the
BINDING_ANCHOR port without any existing binding in the SEND SAVI
device. The SEND SAVI device has sent a NUD_NSOL message to the
BINDING_ANCHOR port. The relevant events for this case are the
reception of a NUD_NADV from the BINDING_ANCHOR port; the reception
of a DAD_NSOL from the BINDING_ANCHOR port, other VP different from
the BINDING_ANCHOR port, or a TP; and the reception of any packet
other than a DAD_NSOL and a NUD_NADV from the BINDING_ANCHOR port and
a DAD_NSOL for other VP different from the BINDING_ANCHOR port, or
TP. In addition, the LIFETIME may expire.

Messages received from the BINDING_ANCHOR port:

o If a validated NUD_NADV message is received through the
   BINDING_ANCHOR port, the LIFETIME is set to TENT_LT, and the state
   is changed to VALID. The message is not forwarded to any port.

o If a validated DAD_NSOL message is received through the
   BINDING_ANCHOR port, it is forwarded to the appropriate Trusted
   ports, the LIFETIME is set to TENT_LT, and the state is changed to
   TENTATIVE_DAD.

o Any packet other than NUD_NADV or DAD_NSOL received through the
   BINDING_ANCHOR port is discarded.

Messages received from a Validating port different from the
BINDING_ANCHOR:

o If a validated DAD_NSOL message is received through port VP'
   different from the BINDING_ANCHOR port, it is forwarded to the
   appropriate Trusted ports, the LIFETIME is set to TENT_LT, the
   BINDING_ANCHOR is set to VP', and the state is changed to
   TENTATIVE_DAD.

o Any packet other than validated DAD_NSOL received through port VP'
   MUST NOT be forwarded unless the next state for the binding is
   VALID. The packets received MAY be discarded or MAY be stored to
   be sent if the state changes later to VALID. The state is left
   unchanged.

Messages received from a Trusted port:

o If a DAD_NSOL message is received through a Trusted port, it is
   forwarded to the BINDING_ANCHOR port, and the state is left
   unchanged.

o Any other packet received from a Trusted port is forwarded
   appropriately. This packet may come from a SEND SAVI device that
   has securely validated the attachment of the node to its
   Validating port, according to SEND SAVI rules. The state is left
   unchanged.

LIFETIME expires:

o If LIFETIME expires, the LIFETIME is cleared and the state is
   changed to NO_BIND.

3.4. SEND SAVI Port Configuration Guidelines

The detailed guidelines for port configuration in SEND SAVI devices
are:

o Ports connected to another SEND SAVI device MUST be configured as
   Trusted ports. Not doing so will prevent off-link traffic from
   being forwarded, along with the following effects for on-link
   traffic: significantly increase the CPU time, memory consumption,
   and signaling traffic due to SEND SAVI validation, in both the
   SEND SAVI devices and the node whose address is being validated.

o Ports connected to hosts SHOULD be configured as Validating ports.
   Not doing so will allow the host connected to that port to send
   packets with a spoofed source address.

   o No more than one host SHOULD be connected to each port.
      Connecting more than one host to a port will allow hosts to
      generate packets with the same source address as the other hosts
      connected to the same port, and will allow replaying attacks to be
      performed as described in Section 5.1.

   o Ports connected to routers MUST be configured as Trusted ports.
      Not doing so results in SEND SAVI devices discarding off-link
      traffic. Note that this means that since routers are connected
      through Trusted ports, they can generate traffic with any source
      address, even those belonging to the link.

   o Ports connected to a chain of one or more legacy switches that
      have other SEND SAVI devices but have no routers or hosts attached
      to them SHOULD be configured as Trusted ports. Not doing so will
      significantly increase the memory consumption in the SEND SAVI
      devices and increase the signaling traffic due to SEND SAVI
      validation.

3.5. VLAN Support

   In the case where the SEND SAVI device is a switch that supports
   customer VLANs [IEEE.802-1Q.2005], the SEND SAVI specification MUST
   behave as if there was one SEND SAVI process per customer VLAN. The
   SEND SAVI process of each customer VLAN will store the binding
   information corresponding to the nodes attached to that particular
   customer VLAN.

3.6. Protocol Constants

   TENT_LT is 500 milliseconds.

   DEFAULT_LT is 5 minutes.

4. Protocol Walk-Through

   In this section, we include two cases that illustrate the behavior of
   SEND SAVI, the change of the attachment port of a host, and the
   attack of a malicious host. We use the topology depicted in
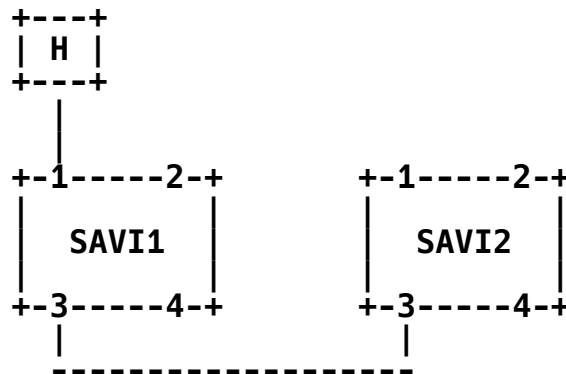   Figure 4.

```
                +---+
                | H |
                +---+
                  |
                  |
          +-1-----2-+        +-1-----2-+
          |         |        |         |
          |  SAVI1  |        |  SAVI2  |
          |         |        |         |
          +-3-----4-+        +-3-----4-+
            |                  |
            ------------------
```

        Figure 4: Reference SEND SAVI Topology for Protocol Walk-Through

4.1. Change of the Attachment Point of a Host

   There are two cases, depending on whether the host H moves to a
   different port on the same switch or to a different switch.

4.1.1. Moving to a Port of the Same Switch

   Host H is connected to port 1 of SAVI1 and moves to port 2 of the
   same switch. Before moving, the SEND SAVI state associated to IPH,
   the IP address of H, is:

   SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=NO_BIND

   In the general case, H issues a DAD_NSOL message for IPH when it is
   connected to a different port. When SAVI1 receives this message, it
   validates it and changes its state to:

   SAVI1=TESTING_VP', BINDING_ANCHOR=1, ALTERNATIVE_BINDING_ANCHOR=2,
   TIMER=TENT_LT / SAVI2=NO_BIND

   The DAD_NSOL message is propagated to port 1, because it is the
   current BINDING_ANCHOR, and the Trusted port 3; it is not propagated
   to Validating port 4. SAVI1 configures a timer for TENT_LT seconds.
   In addition, SAVI1 generates a NUD_NSOL and sends it through port 1.
   When SAVI2 receives this message through its Trusted port, it
   discards it and remains in the NO_BIND state.

SAVI1 waits for a NUD_NADV message to be received from port 1. Since
there is no node attached to 1, there is no response for either of
these messages. When TENT_LT expires at SAVI1, the state changes to:

SAVI1=VALID, BINDING_ANCHOR=2 / SAVI2=NO_BIND

If the node moving does not issue a DAD_NSOL when it attaches to port
2, then SAVI1 will receive a data packet through this port. The data
packet is discarded, SAVI1 issues a secured NUD_NSOL through port 1,
and the state changes to TESTING_VP'.

SAVI1=TESTING_VP', BINDING_ANCHOR=1, ALTERNATIVE_BINDING_ANCHOR=2
TIMER=TENT_LT / SAVI2=NO_BIND

SAVI1 waits for a NUD_NADV message to be received from port 1. Since
there is no node attached to 1, there is no response for neither of
these messages. When TENT_LT expires at SAVI1, the state changes to:

SAVI1=VALID, BINDING_ANCHOR=2 / SAVI2=NO_BIND

An alternative behavior allowed by the specification for the case in
which the host does not issue a DAD_NSOL is that SAVI1 does nothing.
In this case, after some time (bounded by DEFAULT_LT), the switch
will change the state for IPH to TESTING_VP, check if H is still at
port 1 (which it is not), and move the state to NO_BIND. Then, a
packet arriving from port 2 would trigger a process that finishes
with a VALID stated with BINDING_ANCHOR=2.

4.1.2. Moving to a Port of a Different Switch

Host H, connected to port 1 of SAVI1, moves to port 4 of SAVI2.
Before moving, the SEND SAVI state associated to IPH, the IP address
of H, is:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=NO_BIND

If H issues a DAD_NSOL message for IPH when it connects to port 4 of
SAVI2, the state is changed to:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=TENTATIVE_DAD,
BINDING_ANCHOR=4, TIMER=TENT_LT

The DAD_NSOL message is propagated only through the Trusted port of
SAVI2. Then, SAVI1 changes its state as follows:

SAVI1=TESTING_VP, BINDING_ANCHOR=1, TIMER=TENT_LT /
SAVI2=TENTATIVE_DAD, BINDING_ANCHOR=4, TIMER=TENT_LT

SAVI1 propagates the DAD_NSOL message to port 1. Since the only node
that can answer with a secured DAD_NUD has moved, the timer at SAVI2
expires, and SAVI2 changes its state to VALID:

SAVI1=TESTING_VP, BINDING_ANCHOR=1, TIMER=TENT_LT / SAVI2=VALID,
BINDING_ANCHOR=4

Just a very short time after, the timer at SAVI1 expires, and the
state changes to NO_BIND:

SAVI1=NO_BIND / SAVI2=VALID, BINDING_ANCHOR=4

If host H does not send a DAD_NSOL when it moves to SAVI2 but instead
sends a data packet, SAVI2 changes its state to TENTATIVE_NUD:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=TENTATIVE_NUD,
BINDING_ANCHOR=4, TIMER=TENT_LT

SAVI2 issues a secured NUD_NSOL through port 4. H is assumed to have
the address configured (otherwise, it should not have generated a
data packet), so it can respond with a NUD_NADV. When SAVI1 receives
the NUD_NADV and validates it, the state is changed to VALID:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=VALID, BINDING_ANCHOR=4

After some time (bounded by DEFAULT_LT), the state in SAVI1 will
expire, and SAVI1 will perform a check for host H:

SAVI1=TESTING_VP, BINDING_ANCHOR=1, TIMER=TENT_LT / SAVI2=VALID,
BINDING_ANCHOR=4

SAVI1 issues a NUD_NSOL through port 1 for IPH. No response is
received in this case, so SAVI1 changes its state to NO_BIND:

SAVI1=NO_BIND / SAVI2=VALID, BINDING_ANCHOR=4

4.2. Attack of a Malicious Host

Host H is attached to the SEND SAVI infrastructure through port 1 of
SAVI1. We consider that host M starts sending data packets using IPH
(the IP address of H) as the source address, without issuing a
DAD_NSOL (a similar analysis can be done for this case).

4.2.1. M Attaches to the Same Switch as the Victim's Switch

The initial state before the attack of M is:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=NO_BIND

M attaches to port 2 of SAVI1 and starts sending data packets. When
SAVI1 receives the data packet, the packet is discarded. SEND SAVI
may issue a secured NUD_NSOL through port 1 and changes the state to:

SAVI1=TESTING_VP', BINDING_ANCHOR=1, ALTERNATIVE_BINDING_ANCHOR=2,
TIMER=TENT_LT / SAVI2=NO_BIND

Host H is still attached to port 1, so it receives the NUD_NSOL and
responds with a secured NUD_NADV. SAVI1 receives this message,
validates it, and changes its state again to:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=NO_BIND

To prevent the drain of CPU resources in SAVI1, the processing of
further packets received from port 2 may be rate-limited, as
discussed in Section 5.2.

An alternative to the previous behavior is that SAVI1 does nothing
when node M starts sending packets from port 2. In this case, when
the timer to renew the state triggers (this time it's bounded by
DEFAULT_LT), SAVI1 moves the state to TESTING_VP, sends a NUD_NSOL
through port 1, host H responds, and the state remains in VALID for
BINDING_ANCHOR=1. In this way, communication of host H is also
defended.

4.2.2. M Attaches to a Different Switch to the Victim's Switch

The initial state before the attack of M is:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=NO_BIND

M attaches to port 2 of SAVI2 and starts sending data packets. When
SAVI2 receives the data packet, it changes the state to:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=TENTATIVE_DAD,
BINDING_ANCHOR=2, TIMER=TENT_LT

SAVI2 issues a secured NUD_NSOL through port 2. Since M does not own
the IPH CGA, it cannot respond to the message. When the timer
expires, the state is moved back to:

SAVI1=VALID, BINDING_ANCHOR=1 / SAVI2=NO_BIND

To prevent the drain of CPU resources in SAVI2, the processing of
further packets received from port 2 may be rate-limited, as
discussed in Section 5.2.

5. Security Considerations

   SEND SAVI operates only with validated SEND messages to create
   bindings. Note that IPv6 packets generated by non-SEND nodes will be
   discarded by the first SEND SAVI device receiving it. Therefore,
   attackers cannot obtain any benefit by not using SEND. In order to
   perform address validation in a mixed scenario comprising SEND and
   non-SEND devices, a different solution is required, which should be
   addressed in another document.

   Nodes MUST NOT assume that all SEND messages received from a SEND
   SAVI device are validated, since these devices only validate the
   messages strictly required for SEND SAVI operation. Among the number
   of messages that are not validated by SEND SAVI, we can name NUD_NSOL
   messages generated by other nodes and its corresponding NUD_NADV
   responses, or RSOL messages.

   SEND SAVI improves protection compared to conventional SAVI as a
   result of the increased ability of SEND nodes to prove address
   ownership.

   A critical security consideration regarding SEND SAVI deals with the
   need of proper configuration of the roles of the ports in a SEND SAVI
   deployment scenario. Regarding security, the main requirement is
   that ports defining the protected perimeter SHOULD be configured as
   Validating ports. Not doing so will allow an attacker to send
   packets using any source address, regardless of the bindings
   established in other SEND SAVI devices.

5.1. Protection against Replay Attacks

   One possible concern about SEND SAVI is its behavior when an attacker
   tries to forge the identity of a legitimate node by replaying SEND
   messages used by the SEND SAVI specification. An attacker could
   replay any of these messages to interfere with the SEND SAVI
   operation. For example, it could replay a DAD_NSOL message to abort
   the configuration of an address for a legitimate node and to gain the
   right to use the address for DEFAULT_LT seconds.

   We can analyze two different cases when considering SEND SAVI replay
   attacks:

   o When the SEND message replayed is used to create or update binding
      information for SEND SAVI, since the port through which this
      message is received is key to the SEND SAVI operation. SEND SAVI
      creates and maintains bindings as a result of the reception of
      DAD_NSOL messages and of the exchange of NUD_NSOL/NUD_NADV
      messages.

o When the SEND message replayed does not result in the update of
   binding information for SEND SAVI and, thus, is not related to the
   specific port through which it was received. Such situations are
   the reception of CPA messages containing certificates, and the
   processing of an RADV message coming from a Trusted port, which
   can be used in SEND SAVI to populate the SEND SAVI Prefix List.
   In these two cases, the security risks are equivalent to those of
   the SEND operation, i.e., we can consider that the information
   will not be changed by its legitimate sender for the time during
   which the SEND specification allows replaying (which depends on
   the values of TIMESTAMP_FUZZ and TIMESTAMP_DRIFT [RFC3971]).

For replay of messages belonging to the second case, i.e., messages
that do not result in changes in the SEND SAVI binding information,
the security provided by SEND is sufficient. For the replay of
messages belonging to the first case, DAD_NSOL and NUD_NSOL/NUD_NADV
messages, protection results from the behavior of SEND SAVI, as
specified in Section 3.3.2, which restricts the ports to which the
messages involved in SEND SAVI binding updates are disseminated.
SEND SAVI devices only forward these messages to ports for which a
binding to the address being tested by the DAD_NSOL message existed.
Therefore, it is not enough for an attacker to subscribe to a
Solicited Node address to receive DAD_NSOL messages sent to that
address, but the attacker needs to generate a valid DAD_NSOL message
associated to the address for which the binding is being tested,
which is deemed unfeasible [RFC3971].

5.2. Protection against Denial-of-Service Attacks

The attacks against the SEND SAVI device basically consist of making
the SEND SAVI device consume its resources until it runs out of them.
For instance, a possible attack would be to send packets with
different source addresses, making the SEND SAVI device create state
for each of the addresses and waste memory. At some point, the SEND
SAVI device runs out of memory and needs to decide how to react. The
result is that some form of garbage collection is needed to prune the
entries. When the SEND SAVI device runs out of the memory allocated
for the SEND SAVI Database, it is RECOMMENDED that it creates new
entries by deleting the entries with a higher Creation time. This
implies that older entries are preserved and newer entries overwrite
each other. In an attack scenario where the attacker sends a batch
of data packets with different source addresses, each new source
address is likely to rewrite another source address created by the
attack itself. It should be noted that entries are also garbage
collected using the DEFAULT_LT, which is updated by NUD_NSOL/NUD_NADV
exchanges. The result is that in order for an attacker to actually
fill the SEND SAVI Database with false source addresses, it needs to
continuously answer to NUD_NSOL for all the different source

addresses, so that the entries grow old and compete with the
legitimate entries. The result is that the cost of the attack is
highly increased for the attacker.

In addition, it is also RECOMMENDED that a SEND SAVI device reserves
a minimum amount of memory for each available port (in the case where
the port is used as part of the L2 anchor). The REQUIRED minimum is
the memory needed to store four bindings associated to the port,
although it SHOULD be raised if the ratio between the maximum number
of bindings allowed in the device and the number of ports is high.
The motivation for setting a minimum number of bindings per port is
as follows. An attacker attached to a given port of a SEND SAVI
device may attempt to launch a DoS attack towards the SEND SAVI
device by creating many bindings for different addresses. It can do
so by sending DAD_NSOL for different addresses. The result is that
the attack will consume all the memory available in the SEND SAVI
device. The above recommendation aims to reserve a minimum amount of
memory per port, so that nodes located in different ports can make
use of the reserved memory for their port even if a DoS attack is
occurring in a different port.

The SEND SAVI device may store data packets while the address is
being verified, for example, when a DAD_NSOL is lost before arriving
to the SEND SAVI device to which the host attaches; when the host
sends data packets, these data packets may be stored until the SEND
SAVI device verifies the binding by means of a NUD packet exchange.
In this case, the memory for data packet storage may also be a target
of DoS attacks. A SEND SAVI device MUST limit the amount of memory
used to store data packets, allowing the other functions (such as
being able to store new bindings) to have available memory even in
the case of an attack, such as those described above.

It is worth noting that the potential of DoS attacks against the SEND
SAVI network is increased due to the use of costly cryptographic
operations in order to validate the address of the nodes. An
attacker could generate packets using new source addresses in order
to make the closest SEND SAVI device spend CPU time to validate
DAD_NSOL messages or to generate a secure NUD_NSOL. This attack can
be used to drain CPU resources of SEND SAVI devices with a very low
cost for the attacker. In order to solve this problem, rate-limiting
the processing of packets that trigger SEND SAVI events SHOULD be
enforced on a per-port basis.

5.3. Considerations on the Deployment Model for Trust Anchors

   The SEND specification [RFC3971] proposes two deployment models for
   trust anchors: either a centralized model relaying on a globally
   rooted public key infrastructure or a more local, decentralized
   deployment model in which end hosts are configured with a collection
   of public keys that are trusted only on a domain.

   The appeal of a centralized model is the possibility for hosts to use
   SEND to validate routers as they move through links belonging to
   different organizations without additional configuration. However,
   without any further protection, it also enables routers authorized
   with a certificate path rooted on a global trust anchor to appear as
   legitimate routers in a link in which they were not intended to act
   as such. This threat already existed for SEND deployments, for which
   links configured to accept centralized trust anchors may send
   outgoing traffic and use prefix information from alien routers. In a
   SEND SAVI deployment, such routers may be able to deliver off-link
   traffic to any node of the link.

   In order to cope with this threat, SEND SAVI specifies that nodes are
   only allowed to behave as routers if they connect through Trusted
   ports. In particular, RADV messages and traffic with off-link source
   addresses are discarded when received through Validating ports, which
   are the ports intended for non-trusted infrastructure, as moving
   nodes. The protection provided by filtering RADV messages prevents
   SEND nodes from identifying alien routers as legitimate routers, even
   though the trust anchor of these routers is valid.

   Besides, it is worth to say that SEND SAVI supports a decentralized
   deployment model.

5.4. Residual Threats

   SEND SAVI assumes that a host will be able to defend its address when
   the DAD procedure is executed for its addresses, and that it will
   answer to a NUD_NSOL with a NUD_NADV when required. This is needed,
   among other things, to support mobility within a link (i.e., to allow
   a host to detach and reconnect to a different layer-2 anchor of the
   same IP subnetwork, without changing its IP address). If the SEND
   SAVI device does not see the DAD_NADV or the NUD_NADV, it may grant
   the binding to a different binding anchor. This means that if an
   attacker manages to prevent a host from defending its source address,
   it will be able to destroy the existing binding and create a new one,
   with a different binding anchor. An attacker may do so, for example,
   by launching a DoS attack to the host that will prevent it to issue
   proper replies.

5.5. Privacy Considerations

A SEND SAVI device MUST delete binding anchor information as soon as possible (i.e., as soon as the state for a given address is back to NO_BIND), except where there is an identified reason why that information is likely to be involved in the detection, prevention, or tracing of actual source address spoofing. Information about the majority of hosts that never spoof SHOULD NOT be logged.

6. Acknowledgments

Thanks to Jean-Michel Combes, Ana Kukec, Ted Lemon, Adrian Farrel, Barry Leiba, Brian Haberman, Vicent Roca, and Benoit Claise for their reviews and comments on this document. The text has also benefited from feedback provided by Tony Cheneau and Greg Daley.

Marcelo Bagnulo is partly funded by Trilogy 2, a research project supported by the European Commission under its Seventh Framework Program. Alberto Garcia-Martinez was supported, in part, by project TEC2012-38362-C03-01, granted by the Spanish Economy and Competitiveness Ministry.

7. References

7.1. Normative References

[RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
            Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3971]   Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure
            Neighbor Discovery (SEND)", RFC 3971, March 2005.

[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.

7.2. Informative References

[IEEE.802-1Q.2005]
          Institute of Electrical and Electronics Engineers, "IEEE
          Standard for Local and Metropolitan Area Networks /
          Virtual Bridged Local Area Networks", IEEE Standard
          802.1Q, May 2005.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering:
          Defeating Denial of Service Attacks which employ IP Source
          Address Spoofing", BCP 38, RFC 2827, May 2000.

[RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node
          Requirements", RFC 6434, December 2011.

[RFC6620] Nordmark, E., Bagnulo, M., and E. Levy-Abegnoli, "FCFS
          SAVI: First-Come, First-Served Source Address Validation
          Improvement for Locally Assigned IPv6 Addresses", RFC
          6620, May 2012.

[RFC7039] Wu, J., Bi, J., Bagnulo, M., Baker, F., and C. Vogt,
          "Source Address Validation Improvement (SAVI) Framework",
          RFC 7039, October 2013.

Authors' Addresses

    Marcelo Bagnulo
    Universidad 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

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

    Phone: 34 91 6248782
    EMail: alberto@it.uc3m.es
    URI: http://www.it.uc3m.es