J. Halpern
Newbridge
E. Levy-Abegnoli, Ed.
Cisco
July 19, 2015

SAVI for Mixed Address Assignment Methods Scenario
draft-ietf-savi-mix-09

Abstract

In case that multiple IP address assignment methods are allowed in a network, the corresponding SAVI methods should be enabled to prevent spoofing in the network. This document reviews how multiple SAVI methods can coexist in a single SAVI device and collisions are resolved when the same binding entry is discovered by two or more methods.

Table of Contents

1. Introduction 2
2. Requirements Language 3
3. Problem Scope 3
4. Architecture 3
5. Recommendations for preventing collisions 5
6. Resolving binding collisions 5
   6.1. Same Address on Different Binding Anchors 5
      6.1.1. Basic preference 6
      6.1.2. Overwritten preference 6
      6.1.3. Multiple SAVI Device Scenario 7
   6.2. Same Address on the Same Binding Anchor 7
7. Security Considerations 7
8. IANA Considerations 8
9. Acknowledgment 8
10. References 8
   10.1. Normative References 8
   10.2. Informative References 9
Authors' Addresses 9

1. Introduction

There are currently several SAVI documents ([RFC6620], [RFC7513] and [RFC7219]) that describe the different methods by which a switch can discover and record bindings between a node's IP address and a binding anchor and use that binding to perform source address validation. Each of these documents specifies how to learn on-link addresses, based on the method used for their assignment, respectively: StateLess Autoconfiguration (SLAAC), Dynamic Host Control Protocol (DHCP) and Secure Neighbor Discovery (SeND). Each of these documents describes separately how one particular method deals with address collisions (same address, different binding anchor).

While multiple IP assignment methods can be used in the same layer-2 domain, a SAVI device might have to deal with a mix of SAVI methods. The purpose of this document is to provide recommendations to avoid collisions and to review collisions handling when two or more such methods come up with competing bindings.

2. 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 RFC 2119 [RFC2119].

3. Problem Scope

Three different IP address assignment methods have been analyzed for SAVI:

1. StateLess Address AutoConfiguration (SLAAC) - analyzed in SAVI-FCFS[RFC6620]

2. Dynamic Host Control Protocol address assignment (DHCP) - analyzed in SAVI-DHCP[RFC7513]

3. Secure Neighbor Discovery (SeND) address assignment, analyzed in SAVI-SEND[RFC7219]

In addition, there is a fourth method for managing (i.e., creation, management, deletion) a binding on the switch, referred to as "manual". It is based on manual binding configuration and is analyzed in [RFC6620] and [RFC7039].

All combinations of address assignment methods can coexist within a layer-2 domain. A SAVI device MUST implement the corresponding binding setup methods (referred to as a "SAVI method") to enable Source Address Validation. If more than one SAVI method is enabled on a SAVI device, the method is referred to as "mix address assignment method" in this document.

SAVI methods are independent from each other, each one handling its own entries. In the absence of reconciliation, each method will reject packets sourced with an address it did not discovered. To prevent addresses discovered by one method to be filtered out by another, the binding table should be shared by all the solutions. However this could create some conflict when the same entry is discovered by two different methods. The purpose of this document is of two folds: provide recommendations and methods to avoid conflicts, and resolve conflicts if and when they happen. Collisions happening within a given method are outside the scope of this document.

4. Architecture

A SAVI device may enable multiple SAVI methods. This mechanism, called SAVI-MIX, is proposed as a arbiter of the binding generation algorithms, generating the final binding entries as illustrated in

Figure 1. Once a SAVI method generates a candidate binding, it will request SAVI-MIX to set up a corresponding entry in the binding table. Then SAVI-MIX will check if there is any conflict in the binding table. A new binding will be generated if there is no conflict. If there is a conflict, SAVI-MIX will determine whether to replace the existing binding or reject the candidate binding based on the policies specified in Section 6.

The packet filtering will not be performed by each SAVI method separately. Instead, SAVI-MIX will perform filtering based on the entries in the binding table.

```
+--------------------------------------------------------+
|                                                        |
|                                         SAVI Device    |
|                                                        |
|                                                        |
|     +-------+     +-------+     +-------+              |
|     | SAVI  |     | SAVI  |     | SAVI  |              |
|     |       |     |       |     |       |              |
|     | FCFS  |     | DHCP  |     | SEND  |              |
|     +-------+     +-------+     +-------+              |
|         |            |            |   Binding          |
|         |            |            |   setup            |
|         v            v            v   requests         |
|     +-------------------------------+                  |
|     |                               |                  |
|     |           SAVI-MIX            |                  |
```

```
|     |                           |                |
|     +---------------------------+                |
|                  |                               |
|                  v           Final Binding        |
|           +-------------+                         |
|           |   Binding   |                         |
|           |             |                         |
|           |   Table     |                         |
|           +-------------+                         |
|                                                  |
+--------------------------------------------------+
```

Figure 1: SAVI-Mix Architecture

Each entry in the binding table will contain the following fields:

1. IP source address

2. Binding anchor

3. Lifetime

4. Creation time

5. Binding methods: the methods which request the binding setup.

5. Recommendations for preventing collisions

If each solution has a dedicated address space, collisions won't happen. Using non overlapping address space across SAVI solutions is therefore recommended. To that end, one should:

1. DHCP/SLAAC: use non-overlapping prefix for DHCP and SLAAC. Set the A bit in Prefix information option of Router Advertisement for SLAAC prefix, and set the M bit in Router Advertisement for DHCP prefix. For detail explanations on these bits, refer to [RFC4861][RFC4862].

2. SeND/non-SeND: avoid mixed environment (where SeND and non-SeND nodes are deployed) or separate the prefixes announced to SeND

and non-SenD nodes. One way to separate the prefixes is to have the router(s) announcing different (non-overlapping) prefixes to SeND and to non-SeND nodes, using unicast Router Advertisements[RFC6085], in response to SeND/non-SeND Router Solicit.

6. Resolving binding collisions

In situations where collisions could not be avoided, two cases should be considered:

1. The same address is bound on two different binding anchors by different SAVI methods.

2. The same address is bound on the same binding anchor by different SAVI methods.

6.1. Same Address on Different Binding Anchors

This would typically occur in case assignment address spaces could not be separated. For instance, an address is assigned by SLAAC on node X, installed in the binding table using SAVI-FCFS, anchored to "anchor-X". Later, the same address is assigned by DHCP to node Y, and SAVI-DHCP will generate a candidate binding entry, anchored to "anchor-Y".

6.1.1. Basic preference

The SAVI device must decide whom the address should be bound with (anchor-X or anchor-Y in this example). Current standard documents of address assignment methods have implied the prioritization relationship, i.e., first-come.

1. SLAAC: s5.4.5 of [RFC4862]

2. DHCPv4: s3.1-p5 of [RFC2131]

3. DHCPv6: s18.1.8 of [RFC3315]

4. SeND: s8 of [RFC3971]

In the absence of any configuration or protocol hint (see Section 6.1.2) the SAVI device should choose the first-come binding anchor, whether it was learnt from SLAAC, SeND or DHCP.

6.1.2. Overwritten preference

There are two identified exceptions to the general prioritization model, one of them being CGA addresses, another one controlled by the configuration of the switch.

6.1.2.1. CGA preference

When CGA addresses are used, and a collision is detected, preference should be given to the anchor that carries the CGA credentials once they are verified, in particular the CGA parameters and the RSA options. Note that if an attacker was trying to replay CGA credentials, he would then compete on the base of "First-Come, First-Served" (FCFS) principle.

6.1.2.2. configuration preference

For configuration driven exceptions, the SAVI device may allow the configuration of a triplet ("prefix", "anchor", "method") or ("address", "anchor", "method"), where at least one of ("anchor", "method") should be specified. Later, if a DAD message is received with the following conditions verified:

1. The target in the DAD message does not exist in the binding table

2. The target is within configured "prefix" (or equal to "address")

3. The anchor bound to target is different from the configured anchor, when specified

4. The configured method, if any, is different from SAVI-FCFS

the switch should defend the address by responding to the DAD message, with a NA message or an ARP response, on behalf of the target node. Plain NA will be replied to SeND DAD. SeND accepts plain NA for the first time (see s8 of [RFC3971]). The probability for a SeND node to generate a colliding address more than one time is trivial in practice, thus the response can effectively protect an existing binding.

It should not at this point install the entry into the binding table. It will simply prevent the node to assign the address, and will de-facto prioritize the configured anchor. This is especially useful to protect well known bindings such as a static address of a server over anybody, even when the server is down. It is also a way to give priority to a binding learnt from SAVI-DHCP over a binding for the same address, learnt from SAVI-FCFS.

6.1.3. Multiple SAVI Device Scenario

A single SAVI device doesn't have the information of all bound addresses on the perimeter. Therefore it is not enough to lookup local bindings to identify a collision. However, assuming DAD is performed throughout the security perimeter for all addresses regardless of the assignment method, then DAD response will inform all SAVI devices about any collision. In that case, FCFS will apply the same way as in a single switch scenario. If the admin configured on one the switches a prefix (or a single static binding) to defend, the DAD response generated by this switch will also prevent the binding to be installed on other switches of the perimeter.

6.2. Same Address on the Same Binding Anchor

A binding may be set up on the same binding anchor by multiple methods, typically SAVI-FCFS and SAVI-DHCP. If the binding lifetimes obtained from the two methods are different, priority should be given to 1) Manual configuration 2) SAVI-DHCP 3) SAVI-FCFS as the least authoritative. The binding will be removed when the prioritized lifetime expires, even if a less authoritative method had a longer lifetime.

7. Security Considerations

SAVI MIX does not eliminate the security problems of each SAVI method. Thus, the potential attacks, e.g., the DoS attack against the SAVI device resource, can still happen. In deployment, the security threats from each enabled SAVI methods should be prevented by the corresponding proposed solutions in each document.

SAVI MIX is only a binding setup/removal arbitration mechanism. It does not introduce additional security threats only if the principle of decision is reasonable. However, there is a slight problem. SAVI MIX is more tolerant about binding establish than each SAVI method alone. As long as one of the enabled SAVI method generates a binding, the binding will be applied. As a result, the allowed binding number limitation or allowed binding setup rate limitation will be the sum of all the enabled SAVI methods. In deployment, whether a SAVI device is capable to support that resource requirement should be evaluated.

8. IANA Considerations

This memo asks the IANA for no new parameters.

9. Acknowledgment

Thanks to Christian Vogt, Eric Nordmark, Marcelo Bagnulo Braun and Jari Arkko for their valuable contributions.

10. References

10.1. Normative References

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <http://www.rfc-editor.org/info/rfc2131>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <http://www.rfc-editor.org/info/rfc3971>.

[RFC6085] Gundavelli, S., Townsley, M., Troan, O., and W. Dec, "Address Mapping of IPv6 Multicast Packets on Ethernet", RFC 6085, DOI 10.17487/RFC6085, January 2011, <http://www.rfc-editor.org/info/rfc6085>.

[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, DOI 10.17487/RFC6620, May 2012, <http://www.rfc-editor.org/info/rfc6620>.

[RFC7039] Wu, J., Bi, J., Bagnulo, M., Baker, F., and C. Vogt, Ed., "Source Address Validation Improvement (SAVI) Framework", RFC 7039, DOI 10.17487/RFC7039, October 2013, <http://www.rfc-editor.org/info/rfc7039>.

[RFC7219] Bagnulo, M. and A. Garcia-Martinez, "SEcure Neighbor Discovery (SEND) Source Address Validation Improvement (SAVI)", RFC 7219, DOI 10.17487/RFC7219, May 2014, <http://www.rfc-editor.org/info/rfc7219>.

[RFC7513] Bi, J., Wu, J., Yao, G., and F. Baker, "Source Address Validation Improvement (SAVI) Solution for DHCP", RFC 7513, DOI 10.17487/RFC7513, May 2015, <http://www.rfc-editor.org/info/rfc7513>.

10.2. Informative References

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <http://www.rfc-editor.org/info/rfc4862>.

Authors' Addresses

Jun Bi
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084
China

EMail: junbi@tsinghua.edu.cn

Guang Yao
Baidu
Baidu Science and Technology Park, Building 1
Beijing 100193
China

EMail: yaoguang.china@gmail.com

Joel M. Halpern
Newbridge Networks Inc

EMail: jmh@joelhalpern.com

Eric Levy-Abegnoli (editor)
Cisco Systems
Village d'Entreprises Green Side - 400, Avenue Roumanille
Biot-Sophia Antipolis 06410
France

EMail: elevyabe@cisco.com