SAVI
Internet Draft
Intended status: Standard Tracks
Expires: March 2011

J. Bi, J. Wu
CERNET
G. Yao
Tsinghua Univ.
F. Baker
Cisco

September 8, 2010

SAVI Solution for DHCP
draft-ietf-savi-dhcp-06.txt

Status of this Memo

This Internet-Draft will expire on January 8, 2010.

Copyright Notice

Abstract

This document specifies the procedure for creating bindings between a DHCPv4 [RFC2131]/DHCPv6 [RFC3315] assigned source IP address and a binding anchor (refer to [SAVI-framework]) on SAVI (Source Address Validation Improvements) device. The bindings can be used to filter packets generated on the local link with forged source IP address.

Table of Contents

Copyright Notice 2
Abstract 2
1. Introduction 3
2. Conventions used in this document........................ 4
3. Mechanism Overview 4
4. Terminology 4
5. Conceptual Data Structures............................... 4
   5.1. Control Plane Data Structure: Binding State Table(BST).. 4
   5.2. Data Plane Data Structure: Filtering Table(FT).......... 5
6. DHCP Scenario 5
7. Binding Anchor Attributes................................ 6
   7.1. No Attribute 6
   7.2. SAVI-Validation Attribute........................... 6
   7.3. SAVI-DHCP-Trust Attribute........................... 7
   7.4. SAVI-SAVI Attribute................................. 7
   7.5. SAVI-BindRecovery Attribute......................... 7
   7.6. SAVI-ExtSnooping Attribute.......................... 7
8. Binding Set Up 7
   8.1. Rationale 8

8.2. Binding States Description............................. 8
8.3. Events 8
8.3.1. Timer expiration event............................ 8
8.3.2. Control message arriving event.................... 8
8.4. Process of Control Packet Snooping...................... 9
8.4.1. From INIT to other states......................... 9
8.4.1.1. Trigger Event................................. 9
8.4.1.2. Following Actions............................ 10
8.4.2. From START to other states....................... 11
8.4.2.1. Trigger Event................................ 11
8.4.2.2. Following Actions............................ 11
8.4.3. From BOUND to other states....................... 12
8.4.3.1. Trigger Event................................ 12
8.4.3.2. Following Actions............................ 12
8.5. State Machine of DHCP Snooping......................... 12
9. Supplemental Binding Process: Handling Link Topology Change. 13
9.1. Binding Recovery Process............................... 14
9.2. Extended Control Packet Snooping Process............... 15
10. Filtering Specification................................. 16
10.1. Data Packet Filtering................................ 16
10.2. Control Packet Filtering............................. 16
11. Handle Binding Anchor Off-link Event.................... 17
12. Binding Number Limitation............................... 17
13. State Restoration 17
14. Confirm Triggered Binding............................... 18
15. Consideration on Link Layer Routing Complexity.......... 18
16. Duplicate Bindings of Same Address...................... 19
17. Constants 19
18. Security Considerations................................. 19
19. IANA Considerations..................................... 19
20. References 19
20.1. Normative References................................. 19
20.2. Informative References............................... 19
21. Acknowledgments 20
22. Change Log 21

1. Introduction

This document describes the procedure for creating bindings between DHCP assigned addresses and a binding anchor (refer to [savi-framework]). Other related details about this procedure are also specified in this document.

These bindings can be used to filter packets with forged IP address. Section 12 suggests usage of these bindings for common practice. [savi-framework] may specify different usages of binding, depending

on the environment and configuration. The definition and examples of binding anchor is specified in [savi-framework].

The binding process is inspired by the work of IP Source Guard [IP Source Guard].

In a stateless DHCP scenario [RFC3736], DHCP is used to configure other parameters but rather IP address. The address of the client SHOULD be bound based on other SAVI solutions, but rather this solution designed for stateful DHCP.

2. Conventions used in this document

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].

3. Mechanism Overview

The mechanism specified in this document is designed to provide an address level source IP address validation granularity, as a supplement to BCP38 [BCP38]. This mechanism is deployed on the access device (including access switch, wireless access point/controller, etc), and performs mainly DHCP snooping to set up bindings between DHCP assigned IP addresses and corresponding binding anchors. The bindings can be used to validate the source address in the packets.

4. Terminology

Main terms used in this document are described in [savi-framework], [RFC2131] and [RFC3315].

5. Conceptual Data Structures

This section describes the possible conceptual data structures used in this mechanism.

Two main data structures are used to record bindings and their states respectively. There is redundancy between the two structures, for the consideration of separation of data plane and control plane.

5.1. Control Plane Data Structure: Binding State Table (BST)

This table contains the state of binding between source address and binding anchor. Entries are keyed on the binding anchor and source IP address. Each entry has a lifetime field recording the remaining lifetime of the entry, a state field recording the state of the

binding and a field recording other information. The lifetime field is used to help remove expired bindings. The state field is used to identify state. The other field is used to keep temporary information, e.g., the transaction ID in DHCP request. Before a binding is finished, the lease time of the address is also kept in this field because it is improper to keep it in the lifetime field which keeps the lifetime of the binding entry but not the address.

Anchor	Address	State	Lifetime	Other
A	IP_1	Bound	65535	
A	IP_2	Bound	10000	
B	IP_3	_Start	1	

Figure 1 Instance of BST

5.2. Data Plane Data Structure: Filtering Table (FT)

This table contains the bindings between binding anchor and address, keyed on binding anchor and address. This table doesn't contain any state of the binding. This table is only used to filter packets. An Access Control List can be regarded as a practical instance of this table.

Anchor	Address
A	IP_1
A	IP_2

Figure 2 Instance of FT

6. DHCP Scenario

Figure 3 shows the main elements in a DHCP enabled network. At least one DHCP server must be deployed in the network, and DHCP relay may be used to relay message between client and server.

Other address assignment mechanisms may be also used in such network. However, this solution is primarily designed for a pure DHCP scenario, in which only DHCP servers can assign valid global address. In a mixed address assignment scenario where multiple address assignment

methods such as DHCPv6 and SLAAC, or DHCPv4 and manually configured assign addresses that share the common prefix, the SAVI device may need additional state in the state machine to detect and avoid address conflict. The SAVI solution for mixed environment is proposed in a separate document [draft-bi-savi-mixed].

```
                                +--------+
                                | DHCP   |
                                | Server |
                                +--------+
                                    |
                                    |
                                    |
                               +----'-----+
                               |  SAVI    |
                               |  Device  |
                               +-/------/-+
                                 |      |
                            +----\-+   +\-----+
                            |DHCP  |   |Client|
                            |Relay |   |      |
                            +------+   +------+
                          Figure 3 DHCP Scenario
```

7. Binding Anchor Attributes

This section specifies the binding anchor attributes involved in this mechanism.

Binding anchor is defined in the [savi-framework]. Attribute of each binding anchor is configurable. In default, binding anchor has no attribute. A binding anchor MAY be configured to have one or more compatible attributes. However, a binding anchor MAY have no attribute.

7.1. No Attribute

By default, a binding anchor has no attribute. Server type DHCP message from binding anchor with no attribute MUST be dropped. However, other packets SHOULD NOT be dropped.

7.2. SAVI-Validation Attribute

SAVI-Validation attribute is used on binding anchor on which the source addresses are to be validated. The filtering process on binding anchor with such attribute is described in section 13.

7.3. SAVI-DHCP-Trust Attribute

SAVI-DHCP-Trust Attribute is used on binding anchor on the path to a trustable DHCP server/relay.

DHCP server/relay message coming from binding anchor with this attribute will be forwarded.

7.4. SAVI-SAVI Attribute

This attribute is used on binding anchor from which the traffic is not to be checked. All traffic from binding anchor with this attribute will be forwarded without check. Note that DHCP server message and router message will also be trusted.

Through configuring this attribute on binding anchor that joins two or more SAVI devices, SAVI-Validation and SAVI-SAVI attributes implement the security perimeter concept in [savi-framework]. Since no binding entry is needed on such binding anchor, the binding entry resource requirement can be reduced greatly.

This attribute can also be set on other binding anchors if the administrator decides not to validate the traffic from the binding anchor.

This attribute is mutually exclusive with SAVI-Validation.

7.5. SAVI-BindRecovery Attribute

This attribute is used on binding anchor that requires binding recovery described in section 10.1.

This attribute is mutually exclusive with SAVI-SAVI.

7.6. SAVI-ExtSnooping Attribute

This attribute is used on binding anchor that requires extended control packet snooping described in section 10.2.

This attribute is mutually exclusive with SAVI-SAVI.

8. Binding Set Up

This section specifies the procedure of setting up bindings based on control packet snooping. The binding procedure specified here is exclusively designed for binding anchor with SAVI-Validation attribute.

8.1. Rationale

The rationale of this mechanism is that if a node attached to a binding anchor intends to use a valid DHCP address, the DHCP procedure which assigns the address to the node goes first on the same binding anchor. This basis stands when the link layer routing is stable. However, unstable link layer routing may result in that data packet is received from a different binding anchor with the DHCP messages. Infrequent link layer path change can be handled (but not perfectly) by the mechanism described in section 10. Section 15 discusses the situation that link layer routing is naturedly unstable. To handle this situation is above the scope of this document.

8.2. Binding States Description

This section describes the binding states of this mechanism.

INIT The state before a binding has been set up.

START A DHCP request (or a DHCPv6 Confirm, or a DHCPv6 Solicitation with Rapid Commit option) has been received from host, and it may trigger a new binding.

BOUND The address is authorized to the client.

8.3. Events

8.3.1. Timer expiration event

EVE_ENTRY_EXPIRE: The lifetime of an entry expires

8.3.2. Control message arriving events

EVE_DHCP_REQUEST: A DHCP Request message is received from a binding anchor with SAVI-Validation attribute, and the binding entry limit on the binding anchor has not been reached.

EVE_DHCP_CONFIRM: A DHCPv6 Confirm message is received from a binding anchor with SAVI-Validation attribute, and the binding entry limit on the binding anchor has not been reached.

EVE_DHCP_OPTION_RC: A DHCPv6 Solicitation message with Rapid Commit option is received from a binding anchor with SAVI-Validation attribute, and the binding entry limit on the binding anchor has not been reached.

EVE_DHCP_REPLY: A DHCPv4 Acknowledgement or DHCPv6 Reply message is received from a binding anchor with SAVI-DHCP-Trust attribute, and the message should be forwarded to a binding anchor with SAVI-Validation attribute, which has an entry in the state of START. The TID field in the entry matches the TID in the message.

EVE_DHCP_REPLY_NULL: A DHCPv4 Acknowledgement or DHCPv6 Reply message is received from a binding anchor with SAVI-DHCP-Trust attribute, and the message should be forwarded to a binding anchor with SAVI-Validation attribute, which has no entry in the state of START or matches the TID field.

EVE_DHCP_DECLINE: A DHCP Decline message is received from a binding anchor with SAVI-Validation attribute. The message declines an address bound with the binding anchor in state of LIVE or DETECTION or BOUND.

EVE_DHCP_RELEASE: A DHCP Release message is received from a binding anchor with SAVI-Validation attribute. The message releases an address bound with the binding anchor in state of LIVE or DETECTION or BOUND.

EVE_DHCP_REPLY_RENEW: A DHCPv4 Acknowledgement or DHCPv6 Reply message is received, which suggests a new lease time of address in state of BOUND.

8.4. Process of Control Packet Snooping

8.4.1. From INIT to other states

8.4.1.1. Trigger Event

EVE_DHCP_REQUEST, EVE_DHCP_CONFIRM, EVE_DHCP_OPTION_RC, EVE_DHCP_REPLY_NULL.

Note that vulnerability may be caused by DHCP Reply triggered initialization. The binding of assigned address and binding anchor may be threatened if the binding mechanism between binding anchor and link layer address is not secure. If one of the following conditions is satisfied, the security can be ensured.

1. Option 82 is used to keep binding anchor in DHCP Request and Reply, or

2. Unspoofable MAC is used as binding anchor(802.11i,802.1ae/af), or

3. The mapping table from MAC to binding anchor is secure.

It is SUGGESTED not to initialize a binding based on DHCP Reply, until the associated mechanism is also implemented.

8.4.1.2. Following Actions

If the triggering event is EVE_DHCP_REQUEST/EVE_DHCP_OPTION_RC:

The SAVI device MUST forward the message.

The SAVI device MUST generate an entry for the binding anchor in the Binding State Table (BST) and set the state field to START. The lifetime of this entry MUST set to be MAX_DHCP_RESPONSE_TIME. The Transaction ID (Refer to Section 2 in [RFC2131] and Section 4.2 in [RFC3315]) field of the request packet MUST be recorded in the entry, except that the mapping from link layer address to binding anchor is secure as specified in section 9.2.1.1.

Anchor	Address	State	Lifetime	Other
A		START	MAX_DHCP_RESPONSE_TIME	TID

Figure 4 Binding entry in BST on client triggered initialization

The TID is kept as a mediator of assigned address and the binding anchor of requesting node, to assure that the assigned address can be bound with binding anchor secure.

If the triggering event is EVE_DHCP_CONFIRM:

Other than the actions above, the address to be confirmed MUST be recorded in the entry.

Anchor	Address	State	Lifetime	Other
A	Addr	START	MAX_DHCP_RESPONSE_TIME	TID

Figure 5 Binding entry in BST on Confirm triggered initialization

If the triggering event is EVE_DHCP_REPLY_NULL:

The SAVI device MUST deliver the message to the destination.

The SAVI device MUST generate a new entry in BST and FT. The binding anchor in entry is looked up based on the destination link layer address, from mapping table from link layer address to

binding anchor (e.g., the MAC-Port mapping table in case that port is used as binding anchor). The state of the corresponding entry is set to be BOUND. The lifetime of the entry MUST be set to be the lease time.

Anchor	Address	State	Lifetime	Other
A	Addr	BOUND	Lease time	

Figure 6 Binding entry in BST on Reply triggered initialization

Anchor	Address
A	Addr

Figure 7 Binding entry in FT on Reply triggered initialization

8.4.2. From START to other states

8.4.2.1. Trigger Event

EVE_DHCP_REPLY, EVE_ENTRY_EXPIRE.

8.4.2.2. Following Actions

If the trigger event is EVE_DHCP_REPLY:

The SAVI device MUST deliver the message to the destination.

The state of the corresponding entry is changed to be BOUND.

If the Address field is null, the lease time in Reply message MUST be recorded in the entry.

If the Address field is not null, the Reply is in response to a Confirm message. If the Reply message is of Status Code Success, perform the procedure in section 19 to fetch the lease time. Otherwise, delete the entry.

Anchor	Address	State	Lifetime	Other
A	Addr	BOUND	Lease time	

Figure 8 From START to BOUND

A corresponding entry MUST also be generated in FT.

If the trigger event is EVE_ENTRY_EXPIRE:

The entry MUST be deleted from BST.

8.4.3. From BOUND to other states

8.4.3.1. Trigger Event

EVE_ENTRY_EXPIRE, EVE_DHCP_RELEASE, EVE_DHCP_DECLINE, EVE_DHCP_REPLY_RENEW.

8.4.3.2. Following Actions

If the trigger event is EVE_ENTRY_EXPIRE:

Remove the corresponding entry in BST and FT.

If the trigger event is EVE_DHCP_RELEASE or EVE_DHCP_DECLINE:

Remove the corresponding entry in BST and FT. The Release or Decline message MUST be forwarded.

If the trigger event is EVE_DHCP_REPLY_RENEW:

Set the lifetime of the address to be the new lease time.

8.5. State Machine of DHCP Snooping

The main state transits are listed as follows.

State	Event	Action	Next State
INIT	REQ/CFM/RC	Generate entry	START
*INIT	RPL	Generate entry with lease	BOUND
START	RPL	Record lease time	BOUND

START	Timeout	Remove entry	INIT
BOUND	RELEASE/DECLINE	Remove entry	INIT
BOUND	Timeout	Remove entry	INIT
BOUND	RPL_RENEW	Set new lifetime	BOUND

*: optional but NOT SUGGESTED.

REQ: EVE_DHCP_REQUEST

CFM: EVE_DHCP_CONFIRM

RC: EVE_DHCP_OPTION_RC

RPL: EVE_DHCP REPLY

DECLINE: DHCP DECLINE

RELEASE: DHCP RELEASE

RPL_RENEW: EVE_DHCP_RPL_RENEW

Timeout: EVE_ENTRY_EXPIRE

9. Supplemental Binding Process: Handling Link Topology Change

Supplemental binding process is designed to cover conditions that packet is sent by node without previous DHCP procedure sensed by the SAVI device. A typical situation is that the link topology change after the binding has been set up, and then the node will send packet to a different port with the bound port. Another scenario is that a node moves on the local link without re-configuration process, which can be regarded as a special case of link topology change. In DHCP scenario, till this document is finished, link topology change is the only two events that must be handled through this supplemental binding process.

Supplemental binding process is designed to avoid permanent legitimate traffic blocking. It is not supposed to set up a binding whenever a data packet with unbound source address is received. Generally, longer time and more packets are needed to trigger supplemental binding processes.

For implementations that will face the above problem:

1. Binding Recovery Process is a conditional SHOULD. This function SHOULD be implemented if the vendor has such ability, unless the implementation is known to be directly attached to host. If the mechanism is not implemented and managed nodes are not directly attached, permanent blocking will happen until the node is re-configured.

2. Extended Control Packet Snooping Process is a MUST.

Other techniques may be prudently chosen as alternative if found to have equivalent or even better function to avoid permanently blocking after discussion, implementation and deployment.

9.1. Binding Recovery Process

Refer to [draft-baker-savi-one-implementation-approach] for a detailed implementation suggestion. The process specified here can only be enabled in condition that implementation can meet the specified hardware requirements described in [draft-baker-savi-one-implementation-approach].

If a binding anchor is set to have SAVI-BindRecovery attribute, a FIFO queue or register MUST be used to save recently filtered packets. The SAVI device will fetch packet from the queue/register to check the source address can be used by corresponding client on the local link with limited rate:

1. If the address has a local conflict, meaning the DAD on the address fails, the packet MUST be discarded. If the address is not being used, go to the next step.

2.

IPv4 address:

Send a DHCPLEASEQUERY [RFC4388] message querying by IP address to all DHCPv4 servers for IPv4 address or a configured server address. The server addresses may be discovered through DHCPv4 Discovery. If no DHCPLEASEACTIVE message is received, discard the packet; otherwise generate a new binding entry for the address.

IPv6 address:

Send a LEASEQUERY [RFC5007] message querying by IP address to All_DHCP_Relay_Agents_and_Servers multicast address or a configured server address. If no successful LEASEQUERY-REPLY is received, discard the packet; otherwise generate a new binding entry for the

address. The SAVI device may repeat this process if a LEASEQUERY-REPLY with OPTION_CLIENT_LINK is received, in order to set up binding entries for all the address of the client.

This process MUST be rate limited to avoid Denial of Services attack against the SAVI device itself. A constant BIND_RECOVERY_INTERVAL is used to control the frequency. Two data based processes on one binding anchor must have a minimum interval time BIND_RECOVERY_INTERVAL. This constant SHOULD be configured prudently to avoid Denial of Services.

This process is not strict secure. The node with SAVI-BindRecovery binding anchor has the ability to use the address of an inactive node, which doesn't reply to the DAD probe.

In case that the SAVI device is a pure layer-2 device, DHCP Confirm MAY be used to replace the DHCP LEASEQUERY. The security degree may degrade for the address may not be assigned by DHCP server.

This process may fail if any DHCP server doesn't support LEASEQUERY.

9.2. Extended Control Packet Snooping Process

In this snooping process, other than DHCP initialization messages, other types of control packets processed by processor of SAVI device, if the source address is not bound, may trigger the device to perform binding process.

The control messages that MUST be processed include: (1) address resolution Neighbor Solicitation; (2) Neighbor Advertisement; (3) neighbor unreachability detection; (4) Multicast Listener Discovery; (5) Address Resolution Protocol; (6) DHCP Renew/Rebind. Other ICMP messages that may be processed by intermediate device may also trigger the binding process.

The SAVI device MUST first perform DAD to check if the address has a local conflict, and then send DHCP LEASEQUERY or Confirm to recover binding based on DHCP server message.

A minimum time interval EXT_SNOOPING_INTERVAL MUST be set to limit the rate of such triggering process.

Note that this process may not be able to avoid permanent block, in case that only data packets are sent by node. Generally, this mechanism is still practical, because data packet sending without control plane communication is rare and suspicious in reality. Normal

traffic will contain control plane communication packets to help traffic setup and fault diagnosis.

10. Filtering Specification

This section specifies how to use bindings to filter packets.

Filtering policies are different for data packet and control packet. DHCP and ND messages that may cause state transit are classified into control packet. Neighbor Advertisement and ARP Response are also included in control packet, because the Target Address of NA and ARP Response should be checked to prevent spoofing. All other packets are considered to be data packets.

10.1. Data Packet Filtering

Data packets with a binding anchor which has attribute SAVI-Validation MUST be checked.

If the source of a packet associated with its binding anchor is in the FT, this packet SHOULD be forwarded; or else the packet SHOULD be discarded, or alternatively the SAVI SHOULD record this violation.

10.2. Control Packet Filtering

For binding anchors with SAVI-Validation attribute:

Discard/record DHCPv4 Discovery with non-all-zeros source IP address. Discard/record DHCPv4 Request whose source IP address is neither all zeros nor a bound address in FT.

Discard/record DHCPv6 Request whose source is not bound with the corresponding binding anchor in FT. Discard/record DHCPv6 Confirm/ Solicit whose source is not a link local address bound with the corresponding binding anchor in FT. The link layer address may be bound based on SAVI-SLAAC solution or other solutions.

Discard/record other types of DHCP messages whose source is not an address bound with the corresponding binding anchor.

Discard/record IPv6 NS and IPv4 gratuitous ARP whose source is not an address bound with the corresponding binding anchor.

Discard/record NA and ARP Replies messages whose target address and source address are not bound with the corresponding binding anchor.

For other binding anchors:

Discard DHCP Reply/Ack messages not from binding anchor with the SAVI-DHCP-Trust attribute or SAVI-SAVI attribute.

11. Handle Binding Anchor Off-link Event

Port DOWN event MUST be handled if switch port is used as binding anchor. In more general case, if a binding anchor turns off-link, this event MUST be handled.

Whenever a binding anchor with attribute SAVI-Validation turns down, the bindings with the binding anchor MUST be kept for a short time.

To handle movement, if receiving DAD NS/Gra ARP request targeting at the address during the period, the entry MAY be removed.

If the binding anchor turns on-link during the period, recover bindings. It may result in some security problem, e.g., a malicious node immediately associates with the binding anchor got off by a previous node, and then it can use the address assigned to the previous node. However, this situation is very rare in reality. Authors decide not to handle this situation.

12. Binding Number Limitation

It is suggested to configure some mechanism in order to prevent a single node from exhausting the binding table entries on the SAVI device. Either of the following mechanism is sufficient to prevent such attack.

1. Set the upper bound of binding number for each binding anchor with SAVI-Validation.

2. Reserve a number of binding entries for each binding anchor with SAVI-Validation attribute and all binding anchors share a pool of the other binding entries.

3. Limit DHCP Request rate per binding anchor, using the bound entry number of each binding anchor as reverse indicator.

13. State Restoration

If a SAVI device reboots accidentally or designedly, the states kept in volatile memory will get lost. This may cause hosts indirectly attached to the SAVI device to be broken away from the network, because they can't recover bindings on the SAVI device of themselves. Thus, binding entries MUST be saved into non-volatile storage whenever a new binding entry changes to BOUND state or a binding with

state BOUND is removed in condition that this function is supported by hardware. Immediately after reboot, the SAVI device MUST restore binding states from the non-volatile storage. The lifetime and the system time of save process MUST be stored. Then the device MUST check whether the saved entries are obsolete when rebooting.

The possible alternatives proposed but not suitable for general cases are:

If the SAVI device is also the DHCP relay, an alternative mechanism is fetching the bindings through bulk DHCP LEASEQUERY [RFC5460].

If the network enables 802.1ag, the bindings can be recovered with the help of the first hop routers through snooping unicast Neighbor Solicitations sent by routers based on the Neighbor Table.

14. Confirm Triggered Binding

If a binding entry is triggered by a CONFIRM message from the client, no lease time will be contained in the REPLY from DHCP server. The SAVI device MUST send LEASEQUERY message to get the lease time of the address to complete the binding entry. If no successful LEASEQUERY-REPLY is received, the binding entry SHOULD be removed. In this scenario, the address is not regarded as assigned by DHCP, and it MAY be bound through other SAVI solution.

If the confirmed address has local conflict, the Client-ID field of Confirm and LEASEQUERY-REPLY MUST be compared. If they are not match, the new binding entry MUST be deleted.

15. Consideration on Link Layer Routing Complexity

An implicit assumption of this solution is that data packet must arrive at the same binding anchor with the binding anchor that the control packets have arrived at. If this assumption is not valid, this control packet based solution will fail or at least discard legitimate packet. Unfortunately, if the link layer routing between host and SAVI device is inconsistent from time to time, this assumption doesn't stand. Time consistency of link layer routing is not assured by link layer routing protocol. For example, TRILL, a recent link layer routing protocol, is flexible and multiple link layer paths are allowed.

To make the basic assumption stand, the best way is enforcing that there should be only one topology path from downstream host to the SAVI device. For example, SAVI device is directly attached by hosts.

If the assumption doesn't stand, a better solution is requiring inter-operation between SAVI protocol and the link layer routing protocol to make SAVI protocol sensitive to the link layer routing change. This solution is above the scope of this document.

16. Duplicate Bindings of Same Address

Note that the same address may be bound with multiple binding anchors, only if the binding processes are finished on each binding anchor successfully respectively.

This mechanism is designed in consideration that a node may move on the local ink, and a node may have multiple binding anchors.

Note that the local link movement scenario is not handled perfectly. The former binding may not be removed, unless the node is directly attached to the SAVI device. The nodes sharing the same former binding anchor of the moving node have the ability to use its address.

17. Constants

MAX_DHCP_RESPONSE_TIME 120s

BIND_RECOVERY_INTERVAL Device capacity depended and configurable

18. Security Considerations

There is no security consideration currently.

19. IANA Considerations

There is no IANA consideration currently.

20. References

20.1. Normative References

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

20.2. Informative References

[RFC2131] R. Droms, "Dynamic Host Configuration Protocol", RFC2131, March 1997.

[RFC3307] B. Haberman, "Allocation Guidelines for IPv6 Multicast Addresses", RFC3307, August 2002.

[RFC3315] R. Droms, Ed. "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC3315, July 2003.

[RFC4388] R. Woundy and K. Kinnear, "Dynamic Host Configuration Protocol (DHCP) Leasequery", RFC4388, February 2006.

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

[RFC4862] Thomson, S., Narten, T. and Jinmei, T., "IPv6 Stateless Autoconfiguration", RFC4862, September, 2007.

[RFC5007] J. Brzozowski, K. Kinnear, B. Volz, S. Zeng, "DHCPv6 Leasequery", RFC5007, September 2007.

[RFC5227] S. Cheshire, "IPv4 Address Conflict Detection", RFC5227, July 2008.

[IP Source Guard] Cisco, "Network Security Technologies and Solutions", chapter 7, Cisco Press, May 20, 2008.

[draft-baker-savi-one-implementation-approach] F. Baker, "An implementation approach to Source Address Validation", draft-baker-savi-one-implementation-approach-00.

[draft-bi-savi-mixed] Jun Bi, "Mixed scenario analysis and best effort solution", draft-bi-savi-mixed-00.

21. Acknowledgments

Special thanks to Christian Vogt and Joel M. Halpern for careful review and valuation comments on the state machine and text.
Thanks to Marcelo Bagnulo Braun, Eric Levy-Abegnoli, Mark Williams, Erik Nordmark, Mikael Abrahamsson, Alberto Garcia, Jari Arkko, David Harrington, Pekka Savola, Xing Li, Lixia Zhang, Robert Raszuk, Greg Daley, John Kaippallimalil and Tao Lin for their valuable contributions.

Authors' Addresses

Jun Bi
CERNET
Beijing, China
Email: junbi@cernet.edu.cn

Jianping Wu
CERNET
Beijing, China
Email: jianping@cernet.edu.cn

Guang Yao
Network Research Center, Tsinghua University
Beijing 100084, China
Email: yaog@netarchlab.tsinghua.edu.cn

Fred Baker
Cisco Systems
Email: fred@cisco.com

22. Change Log

From 02 to 03: Section 12, data trigger and counter trigger are combined to binding recovery process. The expression "one of MUST" is changed to "conditional MUST. Conditions related with the implementation are specified. Related constants are changed in section 26."

Main changes from 03 to 04:

- Section "Prefix configuration" is removed.
- Section "Supplemental binding process" is modified in requirement level.
- Sub-section 9.1 "Rationale" is added.
- Section "Filtering during Detection" is removed.

- Section "Handling layer 2 path change" is changed to "Consideration on Link layer routing complexity"

- Section "Background and related protocols" is removed.

Main changes from 04 to 05:

- Trigger events are listed explicitly in section 8.

- Dection and Live states are deleted, together with corresponding sections.

Main change from 05 to 06:

- Section 8.1: reference section 20 is changed to section 15.