SAVI
Internet-Draft
Intended status: Standards Track
Expires: November 7, 2013

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

May 6, 2013

SAVI Solution for DHCP
draft-ietf-savi-dhcp-16

Abstract

This document specifies the procedure for creating a binding between a DHCPv4/DHCPv6 assigned IP address and a binding anchor on a SAVI (Source Address Validation Improvements) device. The bindings set up by this procedure can be used to filter out packets with forged source IP address in DHCP scenario. This mechanism is proposed as a complement to ingress filtering to provide finer-grained source IP address validation.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes a fine-grained source IP address validation mechanism. This mechanism creates bindings between addresses assigned to network attachment points by DHCP and suitable binding anchors (refer to Section 3) of the attachments. Then the bindings are used to identify and filter out packets originated from these attachments with forged source IP addresses. In this way, this mechanism can prevent hosts from spoofing IP addresses assigned to the other attachment points. Compared with [BCP38], which provides prefix granularity source IP address validity, this mechanism can benefit the network with finer-grained validity and traceability of source IP addresses.

This mechanism primarily performs DHCP snooping to set up bindings between IP addresses assigned by DHCP and corresponding binding anchors. This binding process is inspired by the work of [BA2007]. Different from [BA2007], which designs specifications about DHCPv4, this mechanism covers the DHCPv6 snooping process, the data snooping process (refer to Section 7), as well as a number of other technical details. Specially, the data snooping process is a data-triggered binding setup procedure designed to avoid permanent block of valid address in case that DHCP snooping is insufficient to set up all the valid bindings.

This mechanism is designed for the stateful DHCP scenario [rfc2131], [rfc3315]. In the stateless DHCP scenario [rfc3736], a client obtains its addresses through some other mechanisms and so the addresses of the client MUST be bound based on other SAVI solutions, for example, SAVI FCFS[savi-fcfs]. Besides, this mechanism is primarily designed for pure DHCP scenarios in which only addresses assigned through DHCP are allowed. However, it does not block any link-local address. It is because link-local addresses are used by DHCPv6 clients before the clients are assigned a DHCPv6 address. Considering that link-local addresses are generally self-generated, and the spoofing of link local address may disturb this mechanism, it is RECOMMENDED to enable a SAVI solution for link-local addresses, e.g., the SAVI-FCFS [savi-fcfs].

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. Terminology

Binding anchor: A "binding anchor" is defined to be a link layer property of network attachment in [savi-framework]. A list of proper binding anchors can be found in Section 3.2 of [savi-framework].

Attribute: A configurable property of each network attachment which indicates the actions to be performed on packets received from the network attachment.

DHCP address: An IP address assigned to an interface via DHCP.

SAVI-DHCP: The name of this SAVI function for DHCP address.

SAVI device: A network device on which this SAVI function is enabled.

Non-SAVI device: A network device on which this SAVI function is not enabled.

DHCP Client-Server message: A message that is sent from a DHCP client to a DHCP server or DHCP servers. Such a message is of one of the following types:

o DHCPv4 Discover: DHCPDISCOVER [rfc2131]

o DHCPv4 Request: DHCPREQUEST generated during SELECTING state [rfc2131]

o DHCPv4 Renew: DHCPREQUEST generated during RENEWING state [rfc2131]

o DHCPv4 Rebind: DHCPREQUEST generated during REBINDING state [rfc2131]

o DHCPv4 Reboot: DHCPREQUEST generated during INIT-REBOOT state [rfc2131]

o DHCPv4 Decline: DHCPDECLINE [rfc2131]

o DHCPv4 Release: DHCPRELEASE [rfc2131]

o DHCPv4 Inform: DHCPINFORM [rfc2131]

o DHCPv6 Request: REQUEST [rfc3315]

o DHCPv6 Solicit: SOLICIT [rfc3315]

- o   DHCPv6 Confirm: CONFIRM [rfc3315]
- o   DHCPv6 Decline: DECLINE [rfc3315]
- o   DHCPv6 Release: RELEASE [rfc3315]
- o   DHCPv6 Rebind: REBIND [rfc3315]
- o   DHCPv6 Renew: RENEW [rfc3315]
- o   DHCPv6 Information-Request: INFORMATION-REQUEST [rfc3315]

DHCP Server-Client message: A message that is sent from a DHCP server to a DHCP client. Such a message is of one of the following types:

- o   DHCPv4 ACK: DHCPACK [rfc2131]
- o   DHCPv4 NAK: DHCPNAK [rfc2131]
- o   DHCPv4 OFFER: DHCPOFFER [rfc2131]
- o   DHCPv6 Reply: REPLY [rfc3315]
- o   DHCPv6 Advertise: ADVERTISE [rfc3315]
- o   DHCPv6 Reconfigure: RECONFIGURE [rfc3315]

Lease time: The lease time in IPv4 [rfc2131] or the valid lifetime in IPv6 [rfc3315].

Binding entry: An 'permit' rule that defines a valid association between an IP address and a binding anchor.

Binding State Table: The data structure that contains all the binding entries.

4. Deployment Scenario and Configuration

4.1. Elements and Scenario

A list of essential elements in a SAVI-DHCP deployment scenario is given as follows:

(1) DHCP server

(2) DHCP client

(3) SAVI device

And there may be following optional elements in a SAVI-DHCP deployment scenario:

(1) DHCP relay

(2) Non-SAVI device

Figure 1 shows a deployment scenario that contains these elements. Note that a physical device can be multiple elements, e.g, a switch can be both a SAVI device and a DHCP relay. In such cases, the links are logic links rather than physical links.

```
                  +--------+     +------------+
                  |DHCP    |-----|  Non-SAVI  |
                  |Server A|     |  Device 1  |
                  +--------+     +-----|------+
                 ......................|............................
                 .                     |                           .
                 . Protection      +---|------+                    .
                 . Perimeter       |  SAVI    |                    .
                 .                 |  Device C|                    .
                 .                 +---|------+                    .
                 .                     |                           .
                 . +----------+    +---|------+       +----------+ .
    downstream   . |  SAVI    |    |  Non SAVI|       |  SAVI    | .
    link    +----.-|  Device A|----|  Device 3|-------|  Device B| .
            |    . +----|--|--+    +----------+       +-|---|----+ .
            |    .      |  +----------+   ............  |   |      .
            |    '..............      |   .          .  |   |      .
            |           |      .      |   .    +--------+   |      .
       +----|-----+  +--|---+  . +----|-+ . +--|---+ .  +---|----+ .
       | Non-SAVI |  |Client|  . |DHCP  | . |Client| .  |DHCP    | .
       | Device 2 |  |A     |  . |Relay | . |B     | .  |Server B| .
       +----------+  +------+  . +------+ . +------+ .  +--------+ .
                               ............          ...............
```

Figure 1: SAVI-DHCP Scenario

4.2. Attribute

As illustrated in Figure 1, an attachment to a SAVI device can be from either a DHCP client, or a DHCP relay/server, or a SAVI device, or a non-SAVI device. Different actions are performed on traffic originated from different elements. To distinguish different types of attachments, an attachment property named 'attribute' is configured on SAVI devices. This section specifies the attributes used by this mechanism.

Before configuration, an attachment is with no attribute. An attachment MAY be configured to have one or more compatible attributes(refer to Section 4.2.6). The attributes of each attachment MUST be configured before this SAVI-DHCP function is enabled on the attachment. The procedure performed by SAVI devices on traffic from each attachment is determined by the attributes set on the attachment.

Particularly, if an attachment has no attribute, no actions will be performed by this SAVI function on traffic from such attachments. For example, in Figure 1, the attachment from the Non-SAVI Device 1 to the SAVI Device B should be configured with no attribute. It means 1) SAVI devices will neither set up bindings for upstream hosts nor check traffic from upstream hosts; 2) SAVI devices will not snoop DHCP messages from upstream devices unless the DHCP-Trust attribute (refer to Section 4.2.2) is set on the corresponding attachment. The reason that DHCP messages from upstream devices are not trusted by default is discussed in Section 12.6.

4.2.1. Trust Attribute

The "Trust Attribute" indicates the packets from the corresponding attachment are completely trustable.

SAVI devices will not set up bindings for attachments with Trust attribute; DHCP messages and data packets from such attachments with this attribute will not be checked. If the DHCP Server-Client messages from attachments with this attribute can trigger the state transitions specified in Section 6 and Section 7, these messages will be handled by the corresponding processes in Section 6 and Section 7.

This attribute is generally configured on the attachments from other SAVI devices. For example, in Figure 1, the attachment from the SAVI Device A to the SAVI Device B and the attachment from the SAVI Device B to the SAVI Device A should be configured with this attribute. Besides, it can be configured on attachments from Non-SAVI devices only if the Non-SAVI devices will not introduce unchecked traffic from DHCP clients. For example, the attachments from Non-SAVI device

3 to SAVI device A, SAVI device B and SAVI device C can be configured with this attribute, only if Non-SAVI device 3 does not have attachment from DHCP clients.

4.2.2. DHCP-Trust Attribute

The "DHCP-Trust Attribute" indicates the DHCP Server-Client messages from the corresponding attachment is trustable.

SAVI devices will forward DHCP Server-Client messages coming from the attachments with this attribute. If the DHCP Server-Client messages can trigger the state transitions, they will be handled by the binding setup processes specified in Section 6 and Section 7.

This attribute is generally used on the direct attachments from the trusted DHCP servers/relays. In Figure 1, the attachment from the DHCP Relay to the SAVI Device B, and the attachment from the DHCP Server B to the SAVI Device B should be configured with this attribute. It is NOT RECOMMENDED to configure this attribute on the indirect attachments from the non-neighboring DHCP servers/relays unless the attachments do not introduce bogus DHCP Server-Client messages. For example, in Figure 1, the attachment from the Non-SAVI Device 1 to the SAVI Device C should not be configured with this attribute. The related security problem is discussed in Section 12.6.

4.2.3. DHCP-Snooping Attribute

The "DHCP-Snooping Attribute" indicates bindings will be set up based on DHCP snooping.

DHCP Client-Server messages from attachments with this attribute will trigger the setup of bindings. SAVI devices will set up bindings on attachments with this attribute based on the DHCP snooping procedure described in Section 6.

DHCP-Snooping attribute is configured on the attachments from DHCP clients. This attribute can be also used on the attachments from downstream Non-SAVI devices which are attached by DHCP clients. In Figure 1, the attachment from the Client A to the SAVI Device A, the attachment from the Client B to the SAVI Device B, and the attachment from the Non-SAVI Device 2 to the SAVI Device A can be configured with this attribute.

4.2.4. Data-Snooping Attribute

The "Data-Snooping Attribute" indicates data packets from the corresponding attachment may trigger binding setup procedure.

Data packets from attachments with this attribute may trigger the setup of bindings. SAVI devices will set up bindings on attachments with this attribute based on the data-triggered process described in Section 7.

If DHCP-Snooping attribute is configured on an attachment, the bindings on this attachment are set up based on DHCP message snooping. However, in some scenarios, a DHCP address may be used by a DHCP client without DHCP address assignment procedure performed on its current attachment. For such attachments, the Data-Snooping process, which is described in Section 7, is necessary. This attribute is configured on such attachments. The usage of this attribute is further discussed in Section 7.

4.2.5. Validating Attribute

The "Validating Attribute" indicates packets from the corresponding attachment will be checked based on binding entries on the attachment.

Packets coming from attachments with this attribute will be checked based on binding entries on the attachment as specified in Section 8.

Validating attribute is configured on the attachments from which the data packets should be checked. For example, the DHCP clients.

4.2.6. Table of Mutual Exclusions

Different types of attributes may indicate mutually exclusive actions on packet. Mutually exclusive attributes MUST NOT be set on the same attachment. The compatibility of different attributes is listed in Figure 2. Note that although Trust and DHCP-Trust are compatible, there is no need to configure DHCP-Trust on an attachment with Trust attribute.

	Trust	DHCP-Trust	DHCP-Snooping	Data-Snooping	Validating
Trust	-	compatible	mutually exclusive	mutually exclusive	mutually exclusive
DHCP-Trust	compatible	-	compatible	compatible	compatible
DHCP-Snooping	mutually exclusive	compatible	-	compatible	compatible
Data-Snooping	mutually exclusive	compatible	compatible	-	compatible
Validating	mutually exclusive	compatible	compatible	compatible	-

Figure 2: Table of Mutual Exclusions

4.3. Perimeter

SAVI-DHCP can provide perimetrical security as SAVI-FCFS (refer to Section 2.5 in [savi-fcfs]). Through configuring attribute of each attachment properly, a perimeter separating untrusted area and trusted area can be formed:

(1) Configure Validating attribute on the attachments of all the DHCP clients. Configure DHCP-Snooping attribute on these attachments.

(2) Configure Validating attribute on the attachments of downstream Non-SAVI devices which are attached by DHCP clients. Configure DHCP-Snooping attribute on these attachments.

(3) Configure Trust attribute on the attachments of other SAVI devices.

(4) If a Non-SAVI device, or a number of connected Non-SAVI devices, have only attachments from SAVI devices or upstream devices, set their attachments to SAVI devices with Trust attribute.

(5) Configure DHCP-Trust attribute on the direct attachments of DHCP relays/servers.

In this way, attachments with Validating attribute (and generally together with attachments of upstream devices) can form a perimeter separating DHCP clients and trusted devices. Data packet check is only performed on the perimeter.

The perimeter is primarily designed for scalability. Each SAVI device only needs to establish bindings for clients directly attached or indirectly attached through Non-SAVI devices. Binding entries for addresses in the network are distributed on SAVI devices. Then the SAVI devices can protect the inside of the perimeter collaboratively, and each SAVI device is not requred to set up bindings for all the addresses assigned in the network.

Particularly, SAVI-DHCP perimeter only contains trusted DHCP servers/ relays inside. The SAVI devices only trust DHCP Server-Client messages originated inside the perimeter. Because bogus DHCP servers are out of the perimeter, the SAVI devices can be protected from fabricated DHCP messages. Note that even if a DHCP server is valid, it may be not contained in the perimeter. For example, in Figure 1, DHCP server A is valid, but it is attached to a Non-SAVI device. The Non-SAVI device may be attached by attackers which generate fabricated DHCP messages. This binding based mechanism may not have the ability to distinguish whether a message received from the attachment of the Non-SAVI device 1 is from DHCP server A or the attackers. If the DHCP server A is contained in the perimeter, the Non-SAVI device 1 will also be contained in the perimter. However, the Non-SAVI device 1 can introduce fabricated DHCP messages into the perimeter. Thus, the DHCP server A cannot be contained in the perimeter. In this case, the SAVI devices can set up bindings for addresses assigned by DHCP server A through snooping the messages relayed by trusted relay in the network. For example, the DHCP relay may relay messages between DHCP server A and the clients in the network, and the SAVI devices can snoop messages from the DHCP relay which is inside the perimeter. The authentication mechanism enforced between the DHCP relay and the DHCP server outside the perimeter can compensate this binding based mechanism.

5. Binding State Table (BST)

Binding State Table is used to contain the bindings between the IP addresses assigned to the attachments and the corresponding binding anchors of the attachments. Each entry of the table, i.e., binding entry, has 5 fields:

- Binding Anchor(Anchor): the binding anchor, i.e., a link-layer property of the attachment.

- IP Address(Address): the IP address assigned to the attachment by DHCP.

- State: the state of the binding. Possible values of this field are listed in Section 6.2 and Section 7.3.

- Lifetime: the remaining seconds of the binding. The Lifetime field counts down automatically.

- TID: the Transaction ID (TID) (refer to [rfc2131] [rfc3315]) of the corresponding DHCP transaction. TID field is used to associate DHCP Server-Client messages with corresponding binding entries.

An instance of this table is shown in Figure 3.

Anchor	Address	State	Lifetime	TID
A	IP_1	BOUND	65535	TID_1
A	IP_2	BOUND	10000	TID_2
B	IP_3	INIT_BIND	1	TID_3

Figure 3: Instance of BST

6. DHCP Snooping Process

This section specifies the process of setting up bindings based on DHCP snooping, named DHCP Snooping Process. This process is illustrated making use of a state machine.

6.1. Rationale

The rationale of the DHCP Snooping Process is that if a DHCP client is legitimate to use a DHCP address, the DHCP address assignment procedure which assigns the IP address to the client must have been performed on the attachment of the client. This basis stands when

the SAVI device is always on the path(s) from the DHCP client to the DHCP server(s)/relay(s). Without considering the movement of DHCP clients, the SAVI device should be the cut node which separates the DHCP clients and the remaining network containing the DHCP server(s)/relay(s). For most of the layer-2 networks whose topologies are simple, it is possible to deploy this SAVI function at proper devices to meet this requirement.

However, a deployment of this SAVI function may not meet the requirement. Besides, the movement of DHCP clients may make bindings are not set up on their new attachments. These exceptions and the solutions are discussed in Section 7.

6.2. Binding States Description

Following binding states present in this process and the corresponding state machine:

NO_BIND: The state before a binding has been set up.

INIT_BIND: A potential binding has been set up.

BOUND: The binding has been set up.

6.3. Events

This section describes events in this process and the corresponding state machine.

6.3.1. Timer Expiration Event

EVE_ENTRY_EXPIRE: The lifetime of a binding entry expires.

6.3.2. Control Message Arriving Events

EVE_DHCP_REQUEST: A DHCPv4 Request or a DHCPv6 Request message is received.

EVE_DHCP_CONFIRM: A DHCPv6 Confirm message is received.

EVE_DHCP_REBOOT: A DHCPv4 Reboot message is received.

EVE_DHCP_REBIND: A DHCPv4 Rebind or a DHCPv6 Rebind message is received.

EVE_DHCP_RENEW: A DHCPv4 Renew or a DHCPv6 Renew message is received.

EVE_DHCP_OPTION_RC: A DHCPv6 Solicitation message with Rapid Commit

option is received.

EVE_DHCP_REPLY_REQUEST: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is an entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Request message, and the assignment is successful. (Note: A DHCPv6 Reply may be a response to DHCPv6 Request, DHCPv6 Confirm, DHCPv6 Renew, DHCPv6 Rebind, DHCPv6 Decline, DHCPv6 Release and DHCPv6 Information-request. Through checking the Status Code option, IA options and other related options in the Reply message, it is possible to determine the purpose of the DHCPv6 Reply. An implementation should refer to [rfc3315] to identify such messages. The processing of the following six events should check the purpose of the DHCP Reply similarly.)

EVE_DHCP_REPLY_REQUEST_NULL: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is no entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Request message, and the assignment is successful.

EVE_DHCP_REPLY_REBIND: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is an entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Rebind message.

EVE_DHCP_REPLY_REBIND_NULL: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is no entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Rebind message.

EVE_DHCP_REPLY_CONFIRM: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is an entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Confirm message.

EVE_DHCP_REPLY_RENEW: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is an entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Renew message.

EVE_DHCP_REPLY_RENEW_NULL: A DHCPv4 ACK or a DHCPv6 Reply message is received, and there is no entry in the BST whose TID field is the same as the message. The DHCPv6 Reply message must be in response to a DHCPv6 Renew message.

EVE_DHCP_DECLINE: A DHCPv4 Decline or a DHCPv6 Decline message is received.

EVE_DHCP_RELEASE: A DHCPv4 Release or a DHCPv6 Release message is received.

EVE_DCHP_LEASEQUERY: A successful DHCPv6 LEASEQUERY_REPLY (refer to section 4.3.3 of [rfc5007]) is received.

Moreover, only if a DHCP message can pass the following checks, the corresponding event is regarded as a valid event:

- o Attribute check: the DHCP Server-Client messages and LEASEQUERY_REPLY should be from attachments with DHCP-Trust attribute; the DHCP Client-Server messages should be from attachments with DHCP-Snooping attribute.

- o Destination check: the DHCP Server-Client messages should be destined to attachments with DHCP-Snooping attribute.

- o Binding anchor check: the DHCP Client-Server messages which may trigger modification or removal of an existing binding entry must have matched binding anchor with the corresponding entry.

- o TID check: the DHCP Server-Client/Client-Server messages must have matched TID with the corresponding entry.

- o Binding limitation check: the DHCP messages must not cause new binding setup on an attachment whose binding entry limitation has been reached. (refer to Section 12.8).

- o Address check: the address of the DHCP messages should pass the check specified in Section 8.2.

On receiving a DHCP message without triggering a valid event, the state will not transit and actions will not be performed.

6.4. State Machine of DHCP Packet Snooping

This section specifies the transits of each state and the corresponding actions.

6.4.1. From NO_BIND to Other States

6.4.1.1. Trigger Event

EVE_DHCP_REQUEST, EVE_DHCP_OPTION_RC, EVE_DHCP_CONFIRM, EVE_DHCP_REBOOT, EVE_DHCP_REBIND, EVE_DHCP_RENEW, EVE_DHCP_REPLY_NULL.

6.4.1.2. Following Actions

If the triggering event is EVE_DHCP_REQUEST/EVE_DHCP_OPTION_RC/ EVE_DHCP_REBOOT:

The SAVI device MUST forward the message.

The SAVI device will generate an entry in the Binding State Table (BST). The Binding anchor field is set to the binding anchor of the attachment from which the message is received. The State field is set to INIT_BIND. The Lifetime field is set to be MAX_DHCP_RESPONSE_TIME. The TID field is set to the TID of the message. If the message is DHCPv4 Request or DHCPv4 Reboot, the Address field can be set to the address to request, i.e., the 'requested IP address'. An example of the entry is illustrated in Figure 4.

Anchor	Address	State	Lifetime	TID
A		INIT_BIND	MAX_DHCP_RESPONSE_TIME	TID

Figure 4: Binding entry in BST on Request/Rapid Commit/Reboot triggered initialization

If the triggering event is EVE_DHCP_CONFIRM/EVE_DHCP_REBIND/ EVE_DHCP_RENEW:

If the message is DHCPv4 Rebind/Renew, the message MUST be forwarded. In DHCPv6, a DHCP client may try to confirm or rebind or renew multiple addresses in a message. In such case, the SAVI device MUST check whether setting up corresponding bindings will make the binding number limitation exceeded. If the limitation will be exceeded, the message MUST be discarded and the following actions will not be performed; or else the message MUST be forwarded.

The SAVI device will generate corresponding entries in the Binding State Table (BST). The Binding anchor field is set to the binding anchor of the attachment from which the message is received. The State field is set to INIT_BIND. The Lifetime field is set to be MAX_DHCP_RESPONSE_TIME. The TID field is set to the TID of the message. The Address field is set to the address to confirm/ rebind(i.e., 'ciaddr' in DHCPv4 Rebind, IPv6 address in IA Address

options of DHCPv6 Confirm/Rebind). An example of the entries is illustrated in Figure 5.

Anchor	Address	State	Lifetime	TID
A	Addr1	INIT_BIND	MAX_DHCP_RESPONSE_TIME	TID
A	Addr2	INIT_BIND	MAX_DHCP_RESPONSE_TIME	TID

Figure 5: Binding entry in BST on Confirm/Rebind/Renew triggered initialization

If the triggering event is EVE_DHCP_REPLY_REQUEST_NULL/EVE_DHCP_REPLY_REBIND_NULL/EVE_DHCP_REPLY_RENEW_NULL:

If the message is DHCPv4 ACK, the message MUST be forwarded. For DHCPv6, a DHCPv6 Reply message may assign multiple addresses to an attachment. In such case, the SAVI device MUST check whether setting up corresponding bindings will make the binding number limitation exceeded. If the limitation will be exceeded, the message MUST be discarded and the following actions will not be performed; or else the message MUST be forwarded.

If the message is DHCPv4 ACK, the SAVI device MUST generate a corresponding entry in the BST.

If the message is DHCPv6 Reply, the SAVI device MUST generate as many new entries in the BST as the number of assign addresses(IPv6 addresses in all the IA Address options of the message). The Binding anchor field is set to the binding anchor of the destined attachment. The State field is set to be BOUND. The Lifetime field is set to the sum of the lease time in Reply message and MAX_DHCP_RESPONSE_TIME. The Address field is set to the assigned address (i.e., 'yiaddr' in DHCPv4 ACK, IPv6 address in IA Address options of DHCPv6 Reply).

An example of the entries is illustrated in Figure 6.

This process is designed to handle the situation that the client moves after sending a Request/Rebind/Renew message. Vulnerability in this process is discussed in Section 12.1.

Anchor	Address	State	Lifetime	TID
A	Addr1	BOUND	Lease time 1 +MAX_DHCP_RESPONSE_TIME	TID
A	Addr2	BOUND	Lease time 2 +MAX_DHCP_RESPONSE_TIME	TID

Figure 6: Binding entry in BST on Reply triggered initialization

6.4.2. From INIT_BIND to Other States

6.4.2.1. Trigger Event

EVE_DHCP_REPLY_REQUEST, EVE_DHCP_REPLY_REBIND, EVE_DHCP_REPLY_CONFIRM, EVE_ENTRY_EXPIRE.

6.4.2.2. Following Actions

If the trigger event is EVE_DHCP_REPLY_REQUEST/EVE_DHCP_REPLY_REBIND/ EVE_DHCP_REPLY_RENEW:

If the message is DHCPv4 ACK, the message MUST be forwarded. In DHCPv6, a DHCPv6 Reply message may assign multiple addresses to an attachment. In such case, the SAVI device MUST check whether setting up corresponding bindings will make the binding number limitation exceeded. If the limitation will be exceeded, the message MUST be discarded and the following actions will not be performed; or else the message MUST be forwarded.

The Address field of the corresponding entry in the BST is set to the address in the message(i.e., 'yiaddr' in DHCPv4 ACK, IPv6 address in IA Address options of DHCPv6 Reply). The Lifetime field is set to the sum of the lease time in Reply message and MAX_DHCP_RESPONSE_TIME. The State field is changed to BOUND. For DHCPv6, if more than one IA Address options is found in the message, corresponding new entries MUST be generated.

An example of the entries is illustrated in Figure 7.

Anchor	Address	State	Lifetime	TID
A	Addr1	BOUND	Lease time+ MAX_DHCP_RESPONSE_TIME	TID
A	Addr2	BOUND	Lease time+ MAX_DHCP_RESPONSE_TIME	TID

Figure 7: From INIT_BIND to BOUND on DHCP Reply in response to Request/Rebind/Renew

If the trigger event is EVE_DHCP_REPLY_CONFIRM:

The DHCP Reply message is in response to a Confirm message. If the Status Code option of the message is not Success (refer to Section 24.4 of [rfc3315]), the message will be forwarded, but no following actions will be performed. If the Status Code option of the message is Success, the state of the corresponding entry is changed to BOUND. Because [rfc3315] does not require lease time of addresses to be contained in the Reply message, the SAVI device MUST send a LEASEQUERY [rfc5007] message querying by IP address to All_DHCP_Relay_Agents_and_Servers multicast address [rfc3315] or a configured server address. The Lifetime of corresponding entries is set to MAX_LEASEQUERY_DELAY.

An example of the entries is illustrated in Figure 8.

The related security problem about DHCPv6 LEASEQUERY is discussed in Section 12.7.

Anchor	Address	State	Lifetime	TID
A	Addr1	BOUND	MAX_LEASEQUERY_DELAY	TID
A	Addr2	BOUND	MAX_LEASEQUERY_DELAY	TID

Figure 8: From INIT_BIND to BOUND on DHCP Reply in response to

Confirm

If the trigger event is EVE_ENTRY_EXPIRE:

The entry MUST be deleted from BST.

Note: If no DHCP Server-Client messages which assign addresses or confirm addresses are received, corresponding entries will expire automatically. Thus, other DHCP Server-Client messages (e.g., DHCPv4 NAK) are not specially processed.

6.4.3. From BOUND to Other States

6.4.3.1. Trigger Event

EVE_ENTRY_EXPIRE, EVE_DHCP_RELEASE, EVE_DHCP_DECLINE, EVE_DHCP_REPLY_REQUEST, EVE_DHCP_REPLY_RENEW, EVE_DHCP_REPLY_REBIND, EVE_DCHP_LEASEQUERY.

6.4.3.2. Following Actions

If the trigger event is EVE_ENTRY_EXPIRE:

Remove the corresponding entry in BST.

If the trigger event is EVE_DHCP_RELEASE/EVE_DHCP_DECLINE:

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

If the trigger event is EVE_DHCP_REPLY_REQUEST/EVE_DHCP_REPLY_RENEW/ EVE_DHCP_REPLY_REBIND:

Set the Lifetime field of the corresponding entries to be the sum of the new lease time and MAX_DHCP_RESPONSE_TIME.

If the trigger event is EVE_DCHP_LEASEQUERY:

Set the Lifetime field to the sum of the lease time in the LEASEQUERY_REPLY message and MAX_DHCP_RESPONSE_TIME.

6.5. Table of State Machine

The main state transits are listed as follows.

State	Event	Action	Next State
NO_BIND	RQ/RC/CF/RE	Generate entry	INIT_BIND
NO_BIND	RPL_NULL	Generate entry with lease	BOUND
INIT_BIND	RPL_RE	Record lease time	BOUND
INIT_BIND	RPL_CF	Send Leasequery	BOUND
INIT_BIND	Timeout	Remove entry	NO_BIND
BOUND	RLS/DCL	Remove entry	NO_BIND
BOUND	Timeout	Remove entry	NO_BIND
BOUND	RPL_RE	Set new lifetime	BOUND
BOUND	LQR	Record lease time	BOUND

Figure 9: Table of Transit

RQ: EVE_DHCP_REQUEST

CF: EVE_DHCP_CONFIRM

RC: EVE_DHCP_OPTION_RC

RE: EVE_DHCP_REBIND/EVE_DHCP_RENEW/EVE_DHCP_REBOOT

RPL_NULL: EVE_DHCP_REPLY_REQUEST_NULL/EVE_DHCP_REPLY_REBIND_NULL/
EVE_DHCP_REPLY_RENEW_NULL

RPL_RE: EVE_DHCP_REPLY_REQUEST/EVE_DHCP_REPLY_REBIND/
EVE_DHCP_REPLY_RENEW

RPL_CF: EVE_DHCP_REPLY_CONFIRM

DCL: EVE_DHCP_DECLINE

RLS: EVE_DHCP_RELEASE

LQR: EVE_DCHP_LEASEQUERY

Timeout: EVE_ENTRY_EXPIRE

```
                             +-------------+
                             |             |
                   /---------|   NO_BIND   |<----------\
                   |  ------>|             |---------\ |
                   |  |      +-------------+         | |EVE_DHCP_RELEASE
 EVE_DHCP_REQUEST  |  |           EVE_DHCP_RENEW_NULL| |EVE_DHCP_DECLINE
 EVE_DHCP_CONFIRM  |  |TIMEOUT    EVE_DHCP_REPLY_NULL| |TIMEOUT
 EVE_DHCP_OPTION_RC|  |          EVE_DHCP_REBIND_NULL| |
 EVE_DHCP_REBIND   |  |                              | |
 EVE_DHCP_RENEW    |  |     EVE_DHCP_REPLY_RENEW     | |
 EVE_DHCP_REBOOT   |  |     EVE_DHCP_REPLY_REBIND    | |
                   v  |     EVE_DHCP_REPLY_REQUEST   v |
            +-------------+ EVE_DHCP_REPLY_CONFIRM +------------+
            |             |                        |            |
            |  INIT_BIND  ------------------------>|    BOUND   |<-\
            |             |                        |            |  |
            +-------------+                        +------------+  |
                                                          |        |
                                                          \--------/
                                              EVE_DHCP_REPLY_RENEW
                                              EVE_LEASEQUERY_REPLY
                                              EVE_DHCP_REPLY_REBIND
                                              EVE_DHCP_REPLY_REQUEST
```

Figure 10: Diagram of Transit

7. Data Snooping Process

7.1. Scenario

The rationale of the DHCP Snooping Process specified in Section 6 is that if a DHCP client is legitimate to use a DHCP address, the corresponding DHCP address assignment procedure must have been finished on the attachment of the DHCP client. This basis stands when the SAVI device is always on the path(s) from the DHCP client to the DHCP server(s)/relay(s). However, there are two exceptions:

o Multiple paths: there are more than one feasible layer-2 paths from the client to the DHCP server/relay, and the SAVI device is not on all of them. The client may get the address through one of the paths not passing by the SAVI device, but packets from the client can travel through the other paths passing by the SAVI device. Because the SAVI device does not snoop the DHCP assignment procedure, the DHCP snooping procedure will not set up the corresponding binding.

- o Dynamic path: there is only one feasible layer-2 path from the client to the DHCP server/relay, but the path is dynamic due to topology change or layer-2 path change. This situation also covers the local-link movement of clients without address confirm/re-configuration process. In such cases, the DHCP snooping process will not set up the corresponding binding.

Data Snooping Process is designed to avoid permanently blocking legitimate traffic in case of these two exceptions. This process is performed on attachments with Data-Snooping attribute. Data packets without matched binding entry may trigger this process to set up bindings.

Snooping data traffic will introduce considerable burden on the processor and ASIC-to-Processor bandwidth of SAVI devices. Considering the overhead of this process, the implementation of this process is a conditional SHOULD. This function SHOULD be implemented unless the implementation is known to be used in the scenarios without the above exceptions. For example, if the implementation is to be used in networks with tree topology and without host local-link movement, there is no need to implement this process in such scenarios.

This process is not supposed to set up a binding whenever a data packet without matched binding entry is received. Instead, unmatched data packets trigger this process with a probability and generally a number of unmatched packets will be discarded before the binding is set up.

7.2. Rationale

This process makes use of duplication detection and DHCP Leasequery to set up bindings. If an address is not used by another client in the network, and the address has been assigned in the network, the address can be bound with the binding anchor of the attachment from which the unmatched packet is received.

The security issues about this process is discussed is Section 12.2.

7.3. Additional Binding States Description

In addition to Section 6.2, new states used in this process are listed here:

DETECTION: The address in the entry is under local duplication detection.

RECOVERY: The SAVI device is querying the assignment and lease time

of the address in the entry through DHCP Leasequery.

7.4. Events

Additional events in this process are described here. Also, if an event will trigger to set up a new binding entry, the binding entry limit on the binding anchor MUST NOT have not been reached.

EVE_DATA_UNMATCH: A data packet without matched binding is received.

EVE_DATA_CONFLICT: ARP Response/Neighbor Advertisement(NA) message against an address in DETECTION state is received.

EVE_DATA_LEASEQUERY:

IPv4: A DHCPLEASEACTIVE message with IP Address Lease Time option is received.

IPv6: A successful LEASEQUERY-REPLY is received.

The triggering packet should pass the following checks to trigger a valid event:

- Attribute check: the data packet should be from attachments with Data-Snooping attribute; the DHCPLEASEACTIVE/LEASEQUERY_REPLY messages should be from attachments with DHCP-Snooping attribute.

- Binding limitation check: the DHCP messages must not cause new binding setup on an attachment whose binding entry limitation has been reached. (refer to Section 12.8).

- Address check: the address of the DHCP/ARP/NA messages should pass the check specified in Section 8.2.

- Interval check: the interval between two successive EVE_DATA_UNMATCH events triggered by an attachment MUST be no smaller than DATA_SNOOPING_INTERVAL.

- TID check: the DHCPLEASEACTIVE/LEASEQUERY-REPLY messages must have matched TID with the corresponding entry.

7.5. State Machine of Binding Recovery Process

Through using additional states, the state machine of this process doesn't conflict the regular process described in Section 6. Thus, it can be implemented separately without changing the state machine in Section 6.

7.5.1. From NO_BIND to Other States

7.5.1.1. Trigger Event

EVE_DATA_UNMATCH.

7.5.1.2. Following Actions

Determine whether to process this event with a probability. The probability can be configured or calculated based on the state of the SAVI device. This probability should be low enough to mitigate the damage from DoS attack against this process.

Create a new entry in the BST. Set the Binding Anchor field to the corresponding binding anchor of the attachment. Set the Address field to be source address of the packet. Set the State field to DETECTION. Set the Lifetime of the created entry to 2*DAD_TIMEOUT.

Check if the address has a local conflict (it violates an address being used by another node):

(1) IPv4 address: send an Address Resolution Protocol (ARP) Request [rfc826]or a ARP probe [rfc5227] on the address; if there is no response message after DAD_TIMEOUT, send another ARP Request or ARP probe;

(2) IPv6 address: perform Duplicate Address Detection (DAD) [rfc4862] on the address; if there is no response message after DAD_TIMEOUT, perform another DAD procedure.

Because the delivery of detection message is unreliable, the detection message is of a certain possibility of not reaching the targeting node. If the targeting node doesn't get the detection message, the address may be bound with a wrong binding anchor in the further stages. This fault may introduce attack against this mechanism. Thus, the detection is performed again if there is no response after the first detection.

The messages MUST NOT be sent to the attachment from which the triggering packet is received.

The packet which triggers this event SHOULD be discarded.

An example of the entry is illustrated in Figure 11.

Anchor	Address	State	Lifetime	TID
A	Addr1	DETECTION	2*DAD_TIMEOUT	

Figure 11: Binding entry in BST on data triggered initialization

7.5.2. **From DETECTION to Other States**

7.5.2.1. **Trigger Event**

EVE_ENTRY_EXPIRE, EVE_DATA_CONFLICT.

7.5.2.2. **Following Actions**

If the trigger event is EVE_ENTRY_EXPIRE:

(1) IPv4 address: Send a DHCPLEASEQUERY [rfc4388] message querying by IP address to all DHCPv4 servers with IP Address Lease Time option (option 51). The server addresses can be found through DHCPv4 Discovery or from configuration. Change the state of the corresponding entry to RECOVERY. Change the lifetime of the entry to be MAX_LEASEQUERY_DELAY.

(2) 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.

Change the state of the corresponding entry to RECOVERY. Change the lifetime of the entry to be MAX_LEASEQUERY_DELAY. The TID field is set to the TID used in the Leasequery message.

An example of the entry is illustrated in Figure 12.

Anchor	Address	State	Lifetime	TID
A	Addr1	RECOVERY	MAX_LEASEQUERY_DELAY	TID

Figure 12: Binding entry in BST on Lease Query

If the trigger event is EVE_DATA_CONFLICT:

Remove the entry.

7.5.3. From RECOVERY to Other States

7.5.3.1. Trigger Event

EVE_ENTRY_EXPIRE, EVE_DATA_LEASEQUERY.

7.5.3.2. Following Actions

If the trigger event is EVE_DATA_LEASEQUERY:

(1) IPv4 address: Check if the 'chaddr' field (hardware address) of the DHCPLEASEACTIVE message matches the hardware address of the triggering message. If the two addresses do not match, the following actions will not be performed. Change the state of the corresponding binding to BOUND. Set life time to the sum of the value encoded in IP Address Lease Time option of the DHCPLEASEACTIVE message and MAX_DHCP_RESPONSE_TIME. Erase the TID field.

(2) IPv6 address: Change the state of the corresponding binding to BOUND. Set the lifetime to the sum of the valid lifetime extracted from OPTION_CLIENT_DATA option in the LEASEQUERY-REPLY message and MAX_DHCP_RESPONSE_TIME. Erase the TID field.

If multiple addresses are specified in the LEASEQUERY-REPLY message, new entries MUST also be created correspondingly on the same binding anchor.

If the trigger event is EVE_ENTRY_EXPIRE:

Remove the entry.

7.5.4. After BOUND

Note that the TID field contains no value after the binding state changes to BOUND. The TID field is recovered from snooping DHCP Renew/Rebind messages. Because TID is used to associate binding entries with messages from DHCP servers, it must be recovered; or else a number of state transits of this mechanism will be not executed normally.

7.5.4.1. Trigger Event

EVE_DHCP_RENEW/EVE_DHCP_REBIND.

7.5.4.2. Following Action

Set the TID field of the corresponding entry to the TID in the triggering message.

7.6. Table of State Machine

The main state transits are listed as follows.

State	Event	Action	Next State
NO_BIND	EVE_DATA_UNMATCH	Duplication detection	DETECTION
DETECTION	Timeout	Send Leasequery	RECOVERY
DETECTION	EVE_DATA_CONFLICT	Remove entry	NO_BIND
RECOVERY	EVE_DATA_LEASEQUERY	Set lease time	BOUND
RECOVERY	Timeout	Remove entry	NO_BIND
BOUND	RENEW/REBIND	Record TID	BOUND

Figure 13: Table of Transit

RENEW: EVE_DHCP_RENEW

REBIND: EVE_DHCP_REBIND

Timeout: EVE_ENTRY_EXPIRE

```
                               +-------------+
                               |             |
                     /---------|   NO_BIND   |<----------\
                     |  ------>|             |           |
                     |  |      +-------------+           |
   EVE_DATA_UNMATCH  |  |EVE_DATA_CONFLICT               |
                     |  |                                |TIMEOUT
                     |  |                                |
                     |  |                                |
                     |  |                                |
                     |  |                                |
                     v  |                                |
              +-------------+        TIMEOUT         +------------+
              |             |                        |            |
              |  DETECTION  ------------------------>|  RECOVERY  |
              |             |                        |            |
              +-------------+                        +------------+
                                      EVE_DATA_LEASEQUERY|
                          /----------\                   |
            EVE_DHCP_RENEW|          |                   |
           EVE_DHCP_REBIND|    +-----v-------+           |
                          |    |             |           |
                          \----|   BOUND     |<----------/
                               |             |
                               +-------------+
```

Figure 14: Diagram of Transit

8. Filtering Specification

This section specifies how to use bindings to filter out spoofing packets.

Filtering policies are different for data packet and control packet. DHCP and NDP (Neighbor Discovery Protocol) [rfc4861] messages that may cause state transit are classified into control packet. Neighbor Advertisement (NA) 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.

8.1. Data Packet Filtering

Data packets from attachment with attribute Validating MUST be checked.

Packet whose source IP address is a link-local address SHOULD be forwarded.

If the source IP address of a packet is not a link-local address, but there is not a matched entry in BST with state BOUND, this packet MUST be discarded. However, the packet may trigger Data Snooping Process if Data-Snooping attribute is set on the attachment.

The SAVI device MAY record any violation.

8.2. Control Packet Filtering

For attachments with Validating attribute:

Discard DHCPv4 Request message whose source IP address is neither all zeros nor a bound address in BST.

Discard DHCPv6 Request message whose source IP address is neither a link-local address nor bound with the corresponding binding anchor in BST.

Discard NDP messages whose source IP address is neither a link-local address nor bound with the corresponding binding anchor. In addition, discard NA message whose target address is neither a link-local address nor bound with the corresponding binding anchor.

Discard ARP messages whose protocol is IP and sender protocol address is neither all zeros address nor bound with the corresponding binding anchor. In addition, discard ARP Reply messages whose target address is not bound with the corresponding binding anchor.

For attachments with other attributes:

Discard DHCP server/relay type message not from attachments with the DHCP-Trust attribute or Trust attribute.

The SAVI device MAY record any violation.

For attachments with no attribute:

No action will be performed on traffic from such attachments.

9. State Restoration

If a SAVI device reboots, the information kept in volatile memory will be lost. This section specifies the restoration of attribute configuration and BST.

9.1. Attribute Configuration Restoration

The lost of attribute configuration will not break the network: no action will be performed on traffic from attachments with no attribute. However, the lost of attribute configuration makes this SAVI function unable to work.

To avoid the loss of binding anchor attribute configuration, the configuration MUST be able to be stored in non-volatile storage. After the reboot of SAVI device, if the configuration of binding anchor attribute can be found in non-volatile storage, the configuration MUST be used.

9.2. Binding State Restoration

The loss of binding state will cause the SAVI devices discard legitimate traffic. Purely using the Data Snooping Process to recover a large number of bindings is of heavy overhead and considerable delay. Thus, to recover bindings from non-volatile storage, as specified below, is RECOMMENDED.

Binding entries MAY be saved into non-volatile storage whenever a new binding entry changes to BOUND state. If a binding with BOUND state is removed, the saved entry MUST be removed correspondingly.

Immediately after reboot, the SAVI device SHOULD restore binding states from the non-volatile storage. The system time of save process MUST be stored. After rebooting, the SAVI device MUST check whether each entry has been obsolete through comparing the saved lifetime and the difference between the current system time and saved system time.

10. Constants

MAX_DHCP_RESPONSE_TIME 120s

DATA_SNOOPING_INTERVAL 60s and configurable

MAX_LEASEQUERY_DELAY 10s

OFFLINK_DELAY 30s

DAD_TIMEOUT 0.5s

11. MLD Consideration

To perform the duplicate detection in Data Snooping Process Section 7, the SAVI device MUST join the Solicited Node Multicast group of the source address of triggering IPv6 data packet whenever performing duplicate detection.

12. Security Considerations

12.1. Security Problem about Binding Setup Triggered by EVE_DHCP_REPLY_NULL

Whenever the triggering event is EVE_DHCP_REPLY_NULL(EVE_DHCP_REPLY_R
EQUEST_NULL/EVE_DHCP_REPLY_REBIND_NULL/EVE_DHCP_REPLY_RENEW_NULL), the SAVI device will try to bind the assigned address with the attachment whose link layer address is the destination link layer address of the message. However, the assigned address could be bound with a wrong attachment if an attacker can pollute the mapping from the link layer address to attachment in the SAVI device.

For example, the SAVI device is a switch and switch port is used as binding anchor. When the SAVI device receives a DHCP Reply with assigned address IP_A and destination link layer address MAC_A, it will check its MAC-to-port table to find the right port. But the MAC-to-port table might be polluted. For example, the requester with MAC_A is attached to Port_A, but an attacker attached to Port_B announces it has MAC_A. If there is no security mechanism used to protect MAC addresses, the SAVI device can bind MAC_A with Port_B. Then the SAVI device will find MAC_A is at Port_B from the polluted MAC-to-port table and it will bind IP_A with Port_B.

Protection from this attack can be ensured by making sure that one of the following conditions is satisfied:

(1) DHCP Option 82 is used to keep binding anchor in DHCP Request and Reply. DHCP Option 82 can be used to keep the circuit information of the client and returned by the DHCP server. Thus, the binding anchor can be determined from the circuit information in the Option. It can be used whenever an implementation doesn't want to create an entry on receiving DHCP Request message.

(2) MAC address is hard to spoof (e.g., 802.11i, 802.1ae/af).

(3) The mapping table from MAC to binding anchor is secure. For example, whenever switch port is used as binding anchor, some security mechanism is used to ensure the mapping from MAC to switch port is secure.

Specially, if the binding anchor is just link layer address, there is no such security problem due to there is no need to map link layer address to binding anchor.

It is RECOMMENDED to implement/enable one of the mechanisms in the SAVI device.

12.2. Security Problems about the Data Snooping Process

There are two security problems about the Data Snooping Process Section 7:

(1) The Data Snooping Process is costly, but an attacker can trigger it simply through sending a number of data packets. To avoid Denial of Services attack against the SAVI device itself, the Data Snooping Process MUST be rate limited. A constant DATA_SNOOPING_INTERVAL is used to control the frequency. Two Data Snooping Processes on one attachment MUST have a minimum interval time DATA_SNOOPING_INTERVAL. This constant SHOULD be configured prudently to avoid Denial of Service attacks.

(2) The Data Snooping Process may set up wrong bindings if the clients do not reply to the detection probes. An attack will pass the duplicate detection if the client assigned the target address does not reply to the detection probes. The DHCP Leasequery procedure performed by the SAVI device just tells whether the address is assigned in the network or not. However, the SAVI device cannot determine whether the address is just assigned to the triggering attachment from the DHCP Leasequery Reply.

12.3. Issues about Leaving Clients

After a binding is set up, the corresponding client may leave its attachment point. It may leave temporarily due to link flapping, or permanently due to it moves to a new attachment point or just leaves the network. Considering the client may be back shortly, the binding should be kept, or else the legtimate traffic from the client will be blocked. However, if the client leaves permanently, it may be insecure to keep the binding. In case that the binding anchor is a property of the attachment point rather than the client, e.g., the switch port, an attacker which is attached to the attachment point of the leaving client can send spoofing packets with the addresses

assigned to the client. Even if the binding anchor is a property of the client, it is a waste of binding resource to keep bindings for left clients.

The following mechanism is designed to handle the leaving of client:

(1) Whenever a client of Validating attribute leaves, a timer of OFFLINK_DELAY is set with the corresponding binding entries.

(2) If receiving DAD Neighbor Solicitation/Gratuitous ARP request targeting at the address during OFFLINK_DELAY, the entries MAY be removed.

(3) If the binding anchor turns on-link during OFFLINK_DELAY, turn off the timer.

In this way, the bindings of a leaving client is kept for OFFLINK_DELAY. In case of link flapping, the client will not be blocked. If the client leaves permanently, the bindings will be removes after OFFLINK_DELAY.

12.4. Duplicate Bindings of the Same Address

The same address may be bound with multiple binding anchors, only if the binding setup processes are finished on each binding anchor successfully. This mechanism is designed in consideration that a client may move on the local link, and a client may have multiple attachments to a SAVI device.

There are two security issues about such a design:

Firstly, due to allowing one address bound with multiple binding anchors, the traceability of address is weakened. An address can be traced to multiple attachments.

Secondly, in the local link movement scenario, the former binding may not be removed and it can be made use of by an attacker sharing the same binding anchor. For example, when switch port is used as binding anchor and the port is shared by an attacker and a client with a hub, the attacker can make use of the address assigned to the client after the client leaves.

12.5. Compatibility with DNA (Detecting Network Attachment)

DNA [rfc4436] [rfc6059] is designed to decrease the handover latency after re-attachment to the same network. DNA mainly relies on performing reachability test through sending unicast Neighbor Solicitation/Router Solicitation/ARP Request message to determine

whether a previously configured address is still valid. Though DNA provides optimization for clients, there is not sufficient information for this mechanism to migrate the previous binding or establish a new binding. If a binding is set up only through snooping the reachability test message, the binding can be invalid. For example, an attacker can perform reachability test with address bound to another client. If binding is migrated to the attacker, the attacker can successful obtain the binding from the victim. Because this mechanism wouldn't set up a binding based on snooping the DNA procedure, it cannot achieve perfect compatibility with DNA. However, it only means the re-configuration of the interface is slowed but not prevented. Details are discussed as follows.

In Simple DNAv6 [rfc6059], the probe is sent with the source address set to a link-local address, and such messages will not be discarded by the policy specified in section Section 8.2. If a client is re-attached to a previous network, the detection will be completed, and the address will be regarded as valid by the client. However, the candidate address is not contained in the probe. Thus, the binding cannot be recovered through snooping the probe. As the client will perform DHCP procedure at the same time, the binding will be recovered from the DHCP Snooping Process. The DHCP Request messages will not be filtered out by this solution as they have link-local source addresses. Before the DHCP procedure is completed, packets will be filtered out by the SAVI device. In another word, if this SAVI function is enabled, Simple DNAv6 will not help reduce the handover latency. If Data-Snooping attribute is configured on the new attachment of the client, the data triggered procedure may reduce the latency.

In DNAv4 [rfc4436], the ARP probe will be discarded because unbound address is used as sender protocol address. As a result, the client will regard the address under detection is valid. However, the data traffic will be filtered. The DHCP Request message sent by the client will not be discarded, because the source IP address field should be all zero as required by [rfc2131]. Thus, if the address is still valid, the binding will be recovered from the DHCP Snooping Process.

12.6. Bogus DHCP Server Threat

DHCP-Trust attribute is designed to prevent attacks from bogus DHCP servers. However, the security is not strict because messages from the valid DHCP server and the bogus DHCP server may arrive at the SAVI device through the same attachment point. As a result, the SAVI device cannot distinguish valid messages from bogus messages. Because the bindings are set up primarily based on DHCP message from DHCP server, bogus DHCP servers can assign invalid addresses to

clients and bindings for these addresses will be set up by the SAVI device.

Thus, each valid DHCP server/relay MUST NOT share a binding anchor with a untrusted device. In case that a binding anchor is shared by a DHCP server/relay and an untrusted device, DHCP-Trust MUST NOT be configured on the corresponding attachment. For example, in Figure 1, if the SAVI device is a switch and the switch port is used as binding anchor, the attachment from Non-SAVI Device 1 to SAVI Device C cannot be configured with DHCP-Trust because the port is shared by DHCP server A and other clients which are untrusted.

12.7. Authentication in DHCPv6 Leasequery

As required in section 5 of RFC5007, DHCPv6 Leasequery 'Should' use IPsec-based authentication specified in the section 21.1 of RFC3315. However, with the deployment of this mechanism, there may be no need to enforce IPSec to perform DHCP Leasequery.

Through containing the DHCP servers in the protection perimeter, the DHCP servers can be protected from spoofing based attacks. Then through checking the source IP address of Leasequery messages, the DHCP server can identify if the messages are from SAVI devices or not. For the SAVI devices, because the perimeter filters out bogus DHCP messages, they can trust the DHCP Leasequery responses. Thus, there is no need to enforce IPSec to validate the DHCP Leasequery messages in this mechanism.

12.8. Binding Number Limitation

A binding entry will cost a certain high-speed memory resource. In general, a SAVI device can only afford a quite limited number of binding entries. In order to prevent an attacker from overloading the resource of the SAVI device, binding entry limit is set on each attachment. The binding entry limit is the upper bound of binding number for each attachment with Validating attribute. No new binding should be set up after the limit has been reached. Besides, if a DHCP Reply assigns more addresses than the remaining binding entry quota of each client, the message will be discarded and no binding will be set up.

12.9. Residual Threats

As described in [savi-framework], this solution cannot strictly prevent spoofing. There are two scenarios in which spoofing can still happen:

(1) The binding anchor is spoofable. If the binding anchor is spoofable, e.g., plain MAC address, an attacker can use forged binding anchor to send packet which will not be regarded as spoofing by SAVI device. Indeed, using binding anchor that can be easily spoofed is more serious than allowing IP spoofing traffic. For example, an attacker can use the binding anchor of another client to get a large number of addresses, and the SAVI device will refuse to set up new binding for the client whenever the binding number limitation has been reached. Thus, it is RECOMMENDED to use strong enough binding anchor, e.g., switch port, secure association in 802.11ae/af and 802.11i.

(2) The binding anchor is shared by more than one clients. If the binding anchor is shared by more than one clients, the clients can spoof the addresses of each other. For example, if switch port is used as binding anchor a number of clients can attach to the same switch port of a SAVI device through a hub. The SAVI device cannot distinguish packets from different clients and thus the spoofing between them will not be detected. A number of the above security problems are caused by sharing binding anchor. Besides, if binding anchor is shared, TID spoofing based attack is possible. Thus, it is RECOMMENDED to use exclusive binding anchor.

13. IANA Considerations

This memo asks the IANA for no new parameters.

Note to RFC Editor: This section will have served its purpose if it correctly tells IANA that no new assignments or registries are required, or if those assignments or registries are created during the RFC publication process. From the authors' perspective, it may therefore be removed upon publication as an RFC at the RFC Editor's discretion.

14. Acknowledgment

Special thanks to Jean-Michel Combes, Christian Vogt, Joel M. Halpern, Eric Levy-Abegnoli, Marcelo Bagnulo Braun, Jari Arkko, Elwyn Davies, Barry Leiba and Alberto Garcia for careful review and valuation comments on the mechanism and text.

Thanks to Mark Williams, Erik Nordmark, Mikael Abrahamsson, David Harrington, Pekka Savola, Xing Li, Lixia Zhang, Bingyang Liu, Duanqi Zhou, Robert Raszuk, Greg Daley, John Kaippallimalil and Tao Lin for their valuable contributions.

This document was generated using the xml2rfc tool.

15. References

15.1. Informative References

[BA2007] Baker, F., "Cisco IP Version 4 Source Guard", IETF Internet draft (work in progress), November 2007.

[BCP38] Paul, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", RFC 2827, BCP 38, May 2000.

[rfc3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, April 2004.

15.2. Normative References

[rfc2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, BCP 14, Match 1997.

[rfc2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[rfc3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[rfc4388] Woundy, R. and K. Kinnear, "Dynamic Host Configuration Protocol (DHCP) Leasequery", RFC 4388, February 2006.

[rfc4436] Aboba, B., Carlson, J., and S. Cheshire, "Detecting Network Attachment in IPv4 (DNAv4)", RFC 4436, March 2006.

[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.

[rfc5007] Brzozowski, J., Kinnear, K., Volz, B., and S. Zeng, "DHCPv6 Leasequery", RFC 5007, September 2007.

[rfc5227] Cheshire, S., "IPv4 Address Conflict Detection", RFC 5227, July 2008.

[rfc6059] Krishnan, S. and G. Daley, "Simple Procedures for Detecting Network Attachment in IPv6", RFC 6059, November 2010.

[rfc826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", RFC 826, November 1982.

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

[savi-framework] Wu, J., Bi, J., Bagnulo, M., Baker, F., and C. Vogt, Ed., "Source Address Validation Improvement Framework", draft-ietf-savi-framework-06 (work in progress), December 2011.

Appendix A. change log

Main changes from 02 to 03:

(1) 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:

(1) Section "Prefix configuration" is removed.

(2) Section "Supplemental binding process" is modified in requirement level.

(3) Sub-section 9.1 "Rationale" is added.

(4) Section "Filtering during Detection" is removed.

(5) Section "Handling layer 2 path change" is changed to "Consideration on Link layer routing complexity"

(6) Section "Background and related protocols" is removed.

Main changes from 04 to 05:

(1) Trigger events are listed explicitly in section 8.

(2) Detection and Live states are deleted, together with corresponding sections.

Main change from 05 to 06:

(1) Section 8.1: reference to section 20 is changed to section 15.

Main changes from 06 to 07:

(1) So many changes in this modification. We suggest to track http://www.ietf.org/mailarchive/web/savi/current/msg01543.ht ml. Changes are made according to the comments.

Main changes from 07 to 08,09:

(1) The modifications are made according to the comments from Jean-Michel Combes.

Main changes from 09 to 11:

(1) DNA issues raised by Jari Arkko

Main changes from 11 to 12:

(1) The modifications are made according to the comments from Eric, http://www.ietf.org/mail-archive/web/savi/current/msg01778.html.

Main changes from 12 to 13:

(1) Main modifications are made based on comments from Elwyn Davies. http://www.ietf.org/mail-archive/web/gen-art/current/msg07297.html.

(2) Other modifications are made based on comments from Barry Leiba.

Main changes from 13 to 14:

(1) A symbol error is corrected.

Main changes from 14 to 15:

(1) In corresponding to "1. Does section 8 describe the mechanism that a SAVI device must perform if it has been unable to snoop the DHCP traffic between a host and a DHCP server? It appears that way in the document, but it would be good to explicitly state that early in the document when the discussion of topologies is being carried out. This becomes important when arbitrary topologies do not provide a means for the SAVI device to eavesdrop on the DHCP traffic." We specified in s7.1 p1 that arbitrary topologies may result in the regular process cannot set up correct bindings. This is also specified in the beginning of s8.

(2) In corresponding to "2. Section 12 refers to the "tentative address multicast group". Do you really mean the Solicited Node Multicast address that is generated from the configured IPv6 unicast address?" Yes. We have changed s12 to "the SAVI device MUST join the Solicited Node Multicast group of the source address of triggering IPv6 data packet whenever performing duplicate detection."

(3) Other modifications are made according to the gen-art review. Refer to http://netarchlab.tsinghua.edu.cn/~yaog/review.txt.

Main changes from 15 to 16:

(1) Main modifications are made according to the second-round gen-art review.

(2) Improve the quality of writing.

Authors' Addresses

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

Email: junbi@tsinghua.edu.cn

Jianping Wu
Tsinghua University
Computer Science, Tsinghua University
Beijing 100084
China

Email: jianping@cernet.edu.cn

Guang Yao
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084
China

Email: yaoguang@cernet.edu.cn

Fred Baker
Cisco Systems
Santa Barbara, CA 93117
United States

Email: fred@cisco.com