SAVI
Internet-Draft
Intended status: Standards Track
Expires: December 28, 2014

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

June 26, 2014

SAVI Solution for DHCP
draft-ietf-savi-dhcp-27

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 is 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.

Table of Contents

1. Introduction

This document describes a fine-grained source IP address validation mechanism. This mechanism creates bindings between IP 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 [RFC2827], 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 procedure which snoops the header of data packet to set up bindings. It is 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]. Stateless DHCP [RFC3736] is out of scope for this document, because it has nothing to do with IP address allocation. A client doing stateless DHCP acquires its IP address(es) using some other mechanism. The appropriate SAVI method must be based on this mechanism. For example, for hosts using Stateless Auto-configuration address, SAVI-FCFS [RFC6620] should be enabled. 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 [RFC6620].

This mechanism works with DHCPv4 and DHCPv6. However, the DHCP address assignment mechanims in IPv4/IPv6 transition scenarios, e.g., [I-D.ietf-dhc-dhcpv4-over-dhcpv6], are beyond the scope of this document.

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 [RFC7039]. A list of proper binding anchors can be found in Section 3.2 of [RFC7039].

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 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 Note: DHCPv4 Request/Renew/Rebind/Reboot messages can be identified based on the Table 4 of [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 (BST): The data structure that contains all the binding entries.

Binding entry limit: The maximum number of binding entries that may be associated with any binding anchor. Limiting the number of binding entries per binding anchor prevents a malicious or malfunctioning node from overloading the binding table on a SAVI device.

Direct attachment: Ideally, a SAVI device should be an access device which is directly attached by hosts. In such case, the hosts are direct attachments of the SAVI device.

Indirect attachment: A SAVI device can be an aggregation device which is connected with a number of access devices, which are attached by hosts. In such case, the hosts are indirect attachments of the SAVI device. Sometimes, it is expressed as "the hosts are indirectly attached to the SAVI device".

Upstream link: Upstream links are links connected to non-SAVI devices from which the valid source address space of traffic contains the prefixes of other networks.

Upstream device: An upstream device is a non-SAVI device associated with an upstream link. For example, the gateway router of the network.

Downstream link: Downstream links are links connected to non-SAVI devices from which the valid source address space of traffic only contains the prefix(es) of the local network.

Downstream device: A downstream device is a non-SAVI device associated with an downstream link. For example, an access switch in the network.

CUT VERTEX: A cut vertex is 'any vertex whose removal increases the number of connected components'. This is a concept in graph theory. This term is used in Section 6.1 to accurately specify the required deployment location of SAVI devices when they only perform the DHCP snooping process.

Identity Association (IA): "A collection of addresses assigned to a client." [RFC3315]

Detection message: a Neighbor Solicitation or ARP message intended to detect a duplicate address by the Data Snooping Process.

DHCP_DEFAULT_LEASE: default lifetime for DHCPv6 address when the binding is triggered by a DHCPv6 Confirm message but a DHCPv6 leasequery exchange [RFC5007] cannot be performed by the SAVI device to fetch the lease.

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. The "Bogus DHCP Server" is only used to help illustrate the case in Section 4.3.3, but not a necessary element.

```
                  +--------+     +------------+    +----------+
                  |DHCP    |-----|  Non-SAVI  |----|Bogus DHCP|
                  |Server A|     |  Device 1  |    |Server    |
                  +--------+     +-----|------+    +----------+
                 ......................|............................
                 .                     |  upstream link            .
                 . 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

Networks are not isolated and traffic from other networks, i.e., transit traffic specified in [RFC6620], may get into the network with SAVI-DHCP deployed through the upstream links. Since SAVI solutions are limited to check traffic generated from local link, SAVI-DHCP is not to set up bindings for addresses assigned in other networks. Thus, SAVI-DHCP will not set up bindings for addresses appearing on upstream links and will not check data traffic from upstream links. This is why to distinguish upstream/downstream links is essential for SAVI-DHCP. The traffic from upstream links should be checked by a prefix granularity source address validation mechanism to avoid spoofing of local addresses from other networks. How to generate and deploy such a mechanism is beyond the scope of this document.

However, traffic from downstream links are generated from local network. For example, a hub, which is attached by some DHCP clients, is on the downstream link of a SAVI device. The traffic from downstream links should be checked by SAVI-DHCP if possible. However, because DHCP clients on the downstream links are indirectly attached, the security problem caused by shared binding anchor, as described in Section 4.3.5, can be introduced.

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 SAVI-DHCP.

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 the SAVI-DHCP function is enabled. The procedure performed by SAVI devices on traffic from each attachment is determined by the attribute(s) set on the attachment.

Particularly, if an attachment has no attribute, data traffic from this attachment will not be checked by SAVI-DHCP and will be forwarded directly. This prevents SAVI-DHCP from causing a break in the network when it is turned on without any binding anchors configured. However, if a binding anchor has no attribute, this means that the SAVI-DHCP-Trust attribute is not present. Because of this, DHCP server-client messages associated to this binding anchor will be discarded. This prevents a host from connecting to an unconfigured binding anchor and acting as a DHCP server. It is SUGGESTED to configure SAVI-DHCP-Trust on necessary binding anchors before turning on the SAVI-DHCP function.

Binding anchors associated with upstream links MAY have no attribute after configuration. For example, in Figure 1, the attachment from the Non-SAVI Device 1 to the SAVI Device C 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 drop DHCP server-client 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 is discussed in Section 4.3.3.

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 B to the SAVI Device C and the attachment from the SAVI Device C to the SAVI Device B 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 A, 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 any indirect attachment point of the non-neighboring DHCP servers and relays, unless all the elements that can be reached through that attachment point can be trusted, i.e., bogus DHCP Server-Client messages will not be generated by these elements. 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. This issue is discussed in Section 4.3.3.

The implementation for DHCPv6 can refer to [I-D.ietf-opsec-dhcpv6-shield] for more details.

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.

Whenever this attribute is set on an attachment, the "Validating Attribute" MUST be set on the same attachment.

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.

Whenever this attribute is set on an attachment, the "Validating Attribute" MUST be set on the same attachment.

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.

This attribute MUST be used together with "DHCP-Snooping Attribute" or "Data-Snooping Attribute".

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

4.3.1. SAVI-DHCP Perimeter Overview

SAVI devices can form a perimeter separating untrusted and trusted areas, similarly to SAVI-FCFS (refer to Section 2.5 of [RFC6620]). Each SAVI device need only establish bindings for a client if it is connected to that client by a link that crosses the perimeter that encloses the SAVI device.

The perimeter is primarily designed for scalability. This has two implications. First, SAVI devices only need to establish bindings for directly attached clients, or clients indirectly attached through non-SAVI device, rather than all the clients in the network. Second, each SAVI device only need to check traffic from clients attached to it, without checking all the traffic passing by.

Consider the example in Figure 1. The protection perimeter is formed by SAVI Device A, B and C. In this case, SAVI device B doesn't create a binding for client A. However, because SAVI device A filters spoofing traffic from client A, SAVI device B can avoid receiving spoofing traffic from client A.

The perimeter in SAVI-DHCP is not only a perimeter for data packets, but also a perimeter for DHCP messages. The placement of DHCP Relay/ Server, which is not involved in [RFC6620] , is related with the construction of the perimeter. The requirement on the placement and configuration of DHCP Relay/Server are discussed in Section 4.3.3.

4.3.2. SAVI-DHCP Perimeter Configuration Guideline

Through configuring attribute of each attachment properly, a perimeter separating untrusted area and trusted area MUST be formed as follows:

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

(2) Configure Validating and DHCP-Snooping attribute on the indirect attachments of all the DHCP clients (i.e., DHCP clients on the downstream links).

(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, set their attachments to SAVI devices with Trust attribute.

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

In this way, the points of attachments with Validating attribute (and generally together with attachments of upstream devices) on SAVI devices can form a perimeter separating DHCP clients and trusted devices. Data packet check is only performed on the perimeter. The perimeter is also a perimeter for DHCP messages. DHCP-Trust attribute is only configured on the inside links of the perimeter. Only DHCP server-client messages originated in the perimeter are trusted.

4.3.3. On the Placement of DHCP Server/Relay

Based on the configuration guideline, it can be found that the SAVI devices only trust DHCP Server-Client messages originated inside the perimeter. It means the trusted DHCP relays/servers must be placed in the perimeter. DHCP server-client messages will be filtered on the perimeter (Note: server-relay messages will not be filtered). In this way, DHCP server-client messages from bogus DHCP servers are filtered on the perimeter, and then the SAVI devices can be protected from forged DHCP messages.

Such a requirement is due to the limitation of this binding based mechanism. This document makes no assumption that the DHCP server-client messages arriving the perimeter from the outside can be trusted. The binding anchor of a trusted remote DHCP server can be shared by a bogus DHCP server. Thus, the SAVI device cannot distinguish bogus and valid DHCP messages only based on the associated binding anchor of DHCP messages in such case.

Note that even if a DHCP server is valid, it may be not contained in the perimeter based on the guideline. 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 forged 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 perimeter. However, the Non-SAVI device 1 can introduce forged 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 (i.e., IPsec, as specified in section 21.1 of [RFC3315]) enforced between the DHCP relay and the DHCP server outside the perimeter can compensate this binding based mechanism. It is SUGGESTED to configure IPsec between the DHCP relay and the DHCP server in such case. If source address validation is enforced in the whole network, which makes the source IP address trustable, the DHCP relay and the DHCP server can simply authenticate the messages from each other based on the source IP address. Nevertheless, it should be noted that the integrity of the messages is not ensured.

Another consideration on the placement is that if the DHCP server/ relay is not inside the perimeter, the SAVI devices may not be able to set up bindings correctly, because the SAVI devices may not be on the path between the clients and the server/relay, or the DHCP messages are encapsulated (e.g., Relay-reply and Relay-forward).

4.3.4. An alternative deployment

In a number of deployment practices, the traffic from the upstream network are all treated as trustable. In such a case, Trust attribute can be set on the upstream link; and if a Non-SAVI device, or a number of connected Non-SAVI devices, have only attachments from SAVI devices and upstream devices, their attachment to SAVI devices can be set Trust attribute. Then an unclosed perimeter will be formed, as illustrate in Figure 3.

```
    |           .             .           Protection |
    |           |             |           Perimeter  |
    |           |             |                      |
    | Upstream  |             | Upstream             |
    | Link      |             | Link                 |
    |           |             |                      |
    |           |             |                      |
    |      +--------+    +--------+    +--------+    |
    |      |SAVI    |----|Non-SAVI|----|SAVI    |    |
    |      |Device  |    |Device  |    |Device  |    |
    |      +--------+    +--------+    +--------+    |
    |           |                           |        |
    \________________________________________________/
                |                           |
                |                           |
           +--------+                  +--------+
           |DHCP    |                  |DHCP    |
           |Client  |                  |Client  |
           +--------+                  +--------+
```

Figure 3: Alternative Perimeter Configuration

To configure such a perimeter, at least the DHCP messages from upstream networks MUST be ensured to be trustable. How to achieve this is beyond the scope of this document.

4.3.5. Considerations on Binding Anchor

The strength of this binding based mechanism depends on the strength of the binding anchor. 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 can lead to worse outcomes than allowing IP spoofing traffic. For example, an attacker can use the binding anchor of another client to bind 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; as a result, even the legitimate clients cannot access the network. Thus, it is RECOMMENDED to use unspoofable binding anchor, e.g., switch port. This document only focuses on switch port as binding anchor. The implications of using other forms of binding anchors should be properly analyzed.

If the binding anchor is shared by more than one clients, the clients can spoof each other addresses. For example, if a 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 anchors. For example, an attacker can send a DHCP Release message to remove the binding of a client sharing the same binding anchor. Thus, it is RECOMMENDED to use exclusive binding anchor.

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.

IA does not present in the BST because the lease of each address in one IA is assigned respectively. Another reason is, when the binding is set up based on data-snooping, IA cannot be recovered from the leasequery protocol. Last reason is there is no IA for DHCPv4.

An instance of this table is shown in Figure 4.

Anchor	Address	State	Lifetime	TID
Port_1	IP_1	BOUND	65535	TID_1
Port_1	IP_2	BOUND	10000	TID_2
Port_2	IP_3	INIT_BIND	1	TID_3

Figure 4: 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 VERTEX whose removal will disjoin the DHCP client and the remaining network containing the DHCP server(s)/relay(s). For most of the 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. For example, there are multiple paths from a DHCP client to the DHCP server and the SAVI device is only on one of them. Then the SAVI device may not be able to snoop the DHCP procedure. Host movement may also make this requirement can not be met. For example, when a DHCP client moves from one attachment to another attachment in the same network, it may not reinitialize its interface or send a Confirm message because of incomplete protocol implementation. Thus, there can be scenarios in which only performing this DHCP snooping process is insufficient to set up bindings for all the valid DHCP addresses. 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_SOLICIT_RC: A DHCPv6 Solicitation message with Rapid Commit option is received.

EVE_DHCP_REPLY: A DHCPv4 ACK or a DHCPv6 Reply message is received.

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_DHCP_LEASEQUERY: A successful DHCPv6 LEASEQUERY_REPLY (refer to section 4.3.3 of [RFC5007]) is received.

Note: the events listed here do not cover all the DHCP messages in section 3. The messages which do not really determine address usage (DHCPv4 Discover, DHCPv4 Inform, DHCPv6 Solicit without Rapid Commit, DHCPv6 Information-Request, DHCPv4 Offer, DHCPv6 Advertise, DHCPv6 Reconfigure), and which are not necessary to snoop (DHCPv4 NAK, refer to section 6.4.2.1), are not included.

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

- 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.

- Destination check: the DHCP Server-Client messages should be destined to attachments with DHCP-Snooping attribute. This check is performed to ensure the binding is set up on the SAVI device which is nearest to the destination client.

- 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.

- TID check: the DHCP Server-Client/Client-Server messages which may cause modification on existing binding entries must have matched TID with the corresponding entry. Note that this check is not performed on Leasequery and Leasequery-reply messages as they are exchanged between the SAVI devices and the DHCP servers. Besides, this check is not performed on DHCP Renew/Rebind messages (Section 6.4.3).

- 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 11.6).

- Address check: the source 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. Note that if a message does not trigger a valid event but it can pass the checks in Section 8.2, it MUST be forwarded.

6.4. The State Machine of DHCP Snooping Process

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

6.4.1. From NO_BIND to INIT_BIND

6.4.1.1. Trigger Events

Trigger events: EVE_DHCP_REQUEST, EVE_DHCP_SOLICIT_RC, EVE_DHCP_CONFIRM, EVE_DHCP_REBOOT.

6.4.1.2. Following Actions

If the triggering event is EVE_DHCP_REQUEST/EVE_DHCP_SOLICIT_RC/ EVE_DHCP_REBOOT:

The SAVI device MUST forward the message.

The SAVI device will generate an entry in the 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 5.

Anchor	Address	State	Lifetime	TID
Port_1		INIT_BIND	MAX_DHCP_RESPONSE_TIME	TID

Figure 5: Binding entry in BST on Request/Rapid Commit/Reboot triggered initialization

If the triggering event is EVE_DHCP_CONFIRM:

The SAVI device MUST forward the message.

The SAVI device will generate corresponding entries in the BST for all the addresses in each the IA option of the Confirm message. 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(es) to confirm. An example of the entries is illustrated in Figure 6.

Anchor	Address	State	Lifetime	TID
Port_1	Addr1	INIT_BIND	MAX_DHCP_RESPONSE_TIME	TID
Port_1	Addr2	INIT_BIND	MAX_DHCP_RESPONSE_TIME	TID

Figure 6: Binding entry in BST on Confirm triggered initialization

6.4.2. From INIT_BIND to Other States

6.4.2.1. Trigger Events

Trigger events: EVE_DHCP_REPLY, EVE_ENTRY_EXPIRE.

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.

6.4.2.2. Following Actions

If the trigger event is EVE_DHCP_REPLY:

The message MUST be forwarded to the corresponding client.

If the message is DHCPv4 ACK, the Address field of the corresponding entry (i.e., the binding entry whose TID is the same of the message) is set to the address in the message(i.e., 'yiaddr' in DHCPv4 ACK). The Lifetime field is set to the sum of the lease time in ACK message and MAX_DHCP_RESPONSE_TIME. The State field is changed to BOUND.

If the message is DHCPv6 Reply, there are following cases:

1. If the status code is not "Success", no modification on corresponding entries will be made. Corresponding entries will expire automatically if no "Success" Reply is received during the lifetime. The entries are not removed immediately due to the client may be able to use the addresses whenever a "Success" Reply is received ("If the client receives any Reply messages that do not indicate a NotOnLink status, the client can use the addresses in the

IA and ignore any messages that indicate a NotOnLink status." [RFC3315]).

2. If the status code is "Success", the SAVI device checks the IA options in the Reply message.

2.1 If there are no IA options in the Reply message, the DHCP Reply message is in response to a Confirm message. The state of the binding entries with matched TID is changed to BOUND. Because [RFC3315] does not require lease time of addresses to be contained in the Reply message, the SAVI device SHOULD send a LEASEQUERY [RFC5007] message querying by IP address to All_DHCP_Servers multicast address [RFC3315] or a list of configured DHCP server addresses. The Leasequery message is generated for each IP address if multiple addresses are confirmed. The Lifetime of corresponding entries is set to 2*MAX_LEASEQUERY_DELAY. If there is no response message after MAX_LEASEQUERY_DELAY, send the LEASEQUERY message again. An example of the entries is illustrated in Figure 7. The related security problem about DHCPv6 LEASEQUERY is discussed in Section 11.5. If the SAVI device does not send the LEASEQUERY message, a pre-configured lifetime DHCP_DEFAULT_LEASE MUST be set on the corresponding entry. (Note: it is SUGGESUTED to use T1 configured on DHCP servers as the DHCP_DEFAULT_LEASE.)

2.2 If there are IA options in the Reply message, the SAVI device checks each IA option. When the first assigned address is found, the Address field of the binding entry with matched TID is set to the address. 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. If there are more than one address assigned in the message, new binding entries are set up for the remaining address assigned in the IA options. An example of the entries is illustrated in Figure 8. SAVI devices do not specially process IA options with NoAddrsAvail status, because there should be no address contained in such IA options.

Note: the SAVI devices do not check if the assigned addresses are duplicated because in SAVI-DHCP scenarios, the DHCP servers are the only source of valid addresses. However, the DHCP servers should be configured to make sure no duplicated addresses are assigned.

Anchor	Address	State	Lifetime	TID
Port_1	Addr1	BOUND	2*MAX_LEASEQUERY_DELAY	TID
Port_1	Addr2	BOUND	2*MAX_LEASEQUERY_DELAY	TID

Figure 7: From INIT_BIND to BOUND on DHCP Reply in response to Confirm

Anchor	Address	State	Lifetime	TID
Port_1	Addr1	BOUND	Lease time+ MAX_DHCP_RESPONSE_TIME	TID
Port_1	Addr2	BOUND	Lease time+ MAX_DHCP_RESPONSE_TIME	TID

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

If the trigger event is EVE_ENTRY_EXPIRE:

The entry MUST be deleted from BST.

6.4.3. From BOUND to Other States

6.4.3.1. Trigger Events

Trigger events: EVE_ENTRY_EXPIRE, EVE_DHCP_RELEASE, EVE_DHCP_DECLINE, EVE_DHCP_REPLY, EVE_DHCP_LEASEQUERY, EVE_DHCP_RENEW, EVE_DHCP_REBIND.

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:

The message MUST be forwarded.

The SAVI device first gets all the addresses ("Requested IP address" in DHCPv4 Decline, "ciaddr" in DHCPv4 Release, addresses in all the IA options of DHCPv6 Decline/Release) to decline/release in the message. Then the corresponding entries MUST be removed.

If the trigger event is EVE_DHCP_RENEW/EVE_DHCP_REBIND:

The message MUST be forwarded.

In such case, a new TID will be used by the client. The TID field of the corresponding entries MUST be set to the new TID. Note that TID check will not be performed on such messages.

If the trigger event is EVE_DHCP_REPLY:

The message MUST be forwarded.

The DHCP Reply messages received in current states should be in response to DHCP Renew/Rebind.

If the message is DHCPv4 ACK, the SAVI device just simply update the binding entry with matched TID, with the Lifetime field set to be the sum of the new lease time and MAX_DHCP_RESPONSE_TIME.

If the message is DHCPv6 Reply, the SAVI device checks each IA Address option in each IA option. If the valid lifetime of an IA address option is 0, the binding entry with matched TID and address is removed. Or else, set the Lifetime field of the binding entry with matched TID and address to be the sum of the new valid lifetime and MAX_DHCP_RESPONSE_TIME.

The SAVI device does not specially process IA options in Reply message with status NoBinding, because no address is contained in such IA options and no actions will be performed.

If the trigger event is EVE_DHCP_LEASEQUERY:

The message MUST be forwarded.

The message should be in response to the Leasequery message sent in Section 6.4.2. The related binding entry can be determined based on the address in the IA Address option in the Leasequery-reply message. The Lifetime field of the corresponding binding entry is set 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. Note that not all the details are specified in the table and the diagram.

State	Event	Action	Next State
NO_BIND	RQ/RC/CF/RE	Generate entry	INIT_BIND
INIT_BIND	RPL	Record lease time (send lease query if no lease)	BOUND
INIT_BIND	Timeout	Remove entry	NO_BIND
BOUND	RLS/DCL	Remove entry	NO_BIND
BOUND	Timeout	Remove entry	NO_BIND
BOUND	RPL	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_SOLICIT_RC

RE: EVE_DHCP_REBOOT

RPL: EVE_DHCP_REPLY

DCL: EVE_DHCP_DECLINE

RLS: EVE_DHCP_RELEASE

LQR: EVE_DHCP_LEASEQUERY

Timeout: EVE_ENTRY_EXPIRE

```
                             +-------------+
                             |             |
                   /---------|   NO_BIND   |<----------\
                   |  ------>|             |           |
                   |  |      +-------------+           |EVE_DHCP_RELEASE
EVE_DHCP_REQUEST   |  |                                |EVE_DHCP_DECLINE
EVE_DHCP_CONFIRM   |  |TIMEOUT                         |TIMEOUT
EVE_DHCP_SOLICIT_RC|  |                                |
EVE_DHCP_REBOOT    |  |                                |
                   |  |                                |
                   |  |                                |
                   v  |                                |
           +-------------+                        +------------+
           |             |      EVE_DHCP_REPLY    |            |
           |  INIT_BIND  ------------------------>|    BOUND   |<-\
           |             |                        |            |  |
           +-------------+                        +------------+  |
                                                         |        |
                                                         \--------/
                                             EVE_DHCP_REPLY
                                             EVE_DHCP_LEASEQUERY
```

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's use of a DHCP address is legitimate, the corresponding DHCP address assignment procedure must have been finished on the attachment of the DHCP client. This is the case stands when the SAVI device is persistently on the path(s) from the DHCP client to the DHCP server(s)/relay(s). However, there are two case when this does not work:

- o Multiple paths: there is more than one feasible layer-2 paths from the client to the DHCP server/relay, and the SAVI device is not on everyone of them. The client may get its address through one of the paths not passing by the SAVI device, but packets from the client can travel through paths that pass through the SAVI device. Because the SAVI device could not snoop the DHCP packet exchange procedure, the DHCP snooping procedure cannot 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 (for example, some link turns broken due to failure or as planned) or layer-2 path change. This situation also covers the local-link movement of clients without address confirm/re-configuration process. For example, a host changes its attached switch port in a very short time. In such cases, the DHCP snooping process will not set up the corresponding binding.

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

Snooping data traffic introduces considerable burden on the processor and ASIC-to-Processor bandwidth of SAVI devices. Because of the overhead of this process, the implementation of this process is a conditional SHOULD. This function SHOULD be enabled 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 intended to set up a binding whenever a data packet without matched binding entry is received. Instead, unmatched data packets trigger this process probabilistically and generally a number of unmatched packets will be discarded before the binding is set up.

7.2. Rationale

This process makes use of NS/ARP 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 11.1.

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 the creation of a new binding entry, the binding entry limit on the binding anchor MUST NOT be exceeded.

EVE_DATA_UNMATCH: A data packet without matched binding is received.

EVE_DATA_CONFLICT: ARP Reply/Neighbor Advertisement(NA) message against an address in DETECTION state is received from a host other than the one for which the entry was added.

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 11.6).

- Address check: For EVE_DATA_LEASEQUERY, the source address of the DHCP Leasequery messages must pass the check specified in Section 8.2. For EVE_DATA_CONFLICT, the source address and target address of the ARP or NA messages must 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.

- Prefix check: the source address of the data packet should be of a valid local prefix, as specified in section 7 of [RFC7039].

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 DETECTION

7.5.1.1. Trigger Event

Trigger event: EVE_DATA_UNMATCH.

7.5.1.2. Following Actions

Make a probabilistic determination whether to act on this event. 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*DETECTION_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 [RFC0826]or a ARP probe [RFC5227] on the address; if there is no response message after DETECTION_TIMEOUT, send another ARP Request or ARP probe;

(2) IPv6 address: send a Neighbor Solicitation message [RFC4861] targeting on the address; if there is no response message after DETECTION_TIMEOUT, send another Neighbor Solicitation message.

Because the delivery of detection message is unreliable, the detection message may fail to reach the targeting node. If there is a node that has the IP address seen in the Data Snooping Process, it may not get the detection messages. This failure mode enables an attack against the Data Snooping Process. 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.

This local conflict process SHOULD be performed. If it is not performed, the state of the entry is set to RECOVERY, the lifetime is set to 2*MAX_LEASEQUERY_DELAY, and the lease query process specified in the following section will be performed directly.

An example of the entry is illustrated in Figure 11.

```
+---------+-------+---------+----------------------+-------+
| Anchor  |Address| State   | Lifetime             |TID    |
+---------+-------+---------+----------------------+-------+
| Port_1  | Addr1 |DETECTION|2*DETECTION_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

Trigger events: 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 each DHCPv4 server with IP Address Lease Time option (option 51). A list of authorized DHCP servers are kept by the SAVI device. The list should be pre-configured or discovered by sending DHCPv4 Discover messages and parsing the replied DHCPv4 Offer messages. Change the state of the corresponding entry to RECOVERY. Change the lifetime of the entry to be 2*MAX_LEASEQUERY_DELAY. The TID field is set to the TID used in the DHCPLEASEQUERY message. If there is no response message after MAX_LEASEQUERY_DELAY, send a DHCPLEASEQUERY to each DHCPv4 server again.

(2) IPv6 address: Send a LEASEQUERY [RFC5007] message querying by IP address to All_DHCP_Relay_Agents_and_Servers multicast address or a list of pre-configured DHCPv6 server addresses. Change the state of the corresponding entry to RECOVERY. Change the lifetime of the entry to be 2*MAX_LEASEQUERY_DELAY. The TID field is set to the TID used in the LEASEQUERY message. If there

is no response message after MAX_LEASEQUERY_DELAY, send the LEASEQUERY message again.

An example of the entry is illustrated in Figure 12.

Anchor	Address	State	Lifetime	TID
Port_1	Addr1	RECOVERY	2*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

Trigger events: EVE_ENTRY_EXPIRE, EVE_DATA_LEASEQUERY.

7.5.3.2. Following Actions

If the trigger event is EVE_DATA_LEASEQUERY:

IPv4 address:

(1) Send an ARP Request with the Target Protocol Address set to the IP address in the corresponding entry. The ARP Request is only sent to the attachment which triggers the binding. If there is no response after DETECTION_TIMEOUT, send another ARP Request. If there is still no response, the following actions will not be performed. If there is only one identical response, get the sender hardware address. Check if the 'chaddr' field (hardware address) of the DHCPLEASEACTIVE message matches the sender hardware address. If the two addresses do not match, the following actions will not be performed. If there is more than one response, if any of the sender hardware addresses matches the 'chaddr' field (hardware address) of the DHCPLEASEACTIVE message, the following actions are to be performed.

(2) 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.

IPv6 address:

(1) Send a Neighbor Solicitation message with the target address set to the IP address in the corresponding entry. The Neighbor Solicitation is only sent to the attachment which triggers the binding. If there is no response after DETECTION_TIMEOUT, send another Neighbor Solicitation. If there is still no response, the following actions will not be performed.

(2) 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.

(3) After the above checks, if multiple addresses are specified in the LEASEQUERY-REPLY message and there are no corresponding binding entries, new entries MUST also be created correspondingly on the same binding anchor.

If responses are received from multiple DHCP servers, the conflict resolution mechanisms specified in section 6.8 of [RFC4388] and section 4.3.4 of [RFC5007] will be used to determine which message should be used.

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

Trigger events: 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 or NO_BIND
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   |<--------\
                     |  ------>|             |         | TIMEOUT
                     |  |      +-------------+         |(2nd LQ_DELAY)
   EVE_DATA_UNMATCH  |  |                              |
                     |  |                              |
   1st               |  |                              |
   DETECTION_TIMEOUT |  |                              | 1st LQ_DELAY
         /------\    |  |                              |    /---------\
         |      |    |  | EVE_DATA_CONFLICT            |    |         |
         |      v    v  |                              |    v         |
         |    +-------------+        TIMEOUT         +------------+   |
         |    |             |(2nd DETECTION_TIMEOUT) |            |   |
         \----|  DETECTION  ------------------------>|  RECOVERY  ----/
              |             |                        |            |
              +-------------+                        +------------+
                                      EVE_DATA_LEASEQUERY|
                          /----------\  (+ 2x DETECTION) |
            EVE_DHCP_RENEW|          |                   |
           EVE_DHCP_REBIND|    +-----v-------+           |
                          |    |             |           |
                          \----|   BOUND     |<----------/
                               |             |
                               +-------------+
```

Figure 14: Diagram of Transit

LQ_DELAY: MAX_LEASEQUERY_DELAY

8. Filtering Specification

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

Filtering policies are different for data packets and control packets. DHCP and NDP (Neighbor Discovery Protocol) [RFC4861] messages that may cause state transit are classified as control packet. Neighbor Advertisement (NA) and ARP Reply are also included in control packet because the Target Address of NA and ARP Reply should be checked to prevent spoofing. All other packets are classified as data packets.

8.1. Data Packet Filtering

Data packets from attachments with the Validating attribute MUST be checked.

Packet whose source IP address is a link-local address will not be checked. Note: as explained in Section 1, a SAVI solution for link-local addresses, e.g., the SAVI-FCFS [RFC6620], can be enabled to check packets with link-local source address.

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 Section 7 if Data-Snooping attribute is set on the attachment.

Data packets from attachments with no attribute will forwarded without checking.

The SAVI device MAY record any violation.

8.2. Control Packet Filtering

For attachments with the Validating attribute:

DHCPv4 Client-Server messages where source IP address is neither all zeros nor bound with the corresponding binding anchor in the BST MUST be discarded.

DHCPv6 Client-Server messages where source IP address is neither a link-local address nor bound with the corresponding binding anchor in the BST MUST be discarded.

NDP messages where source IP address is neither a link-local address nor bound with the corresponding binding anchor MUST be discarded.

NA messages where target address is neither a link-local address nor bound with the corresponding binding anchor MUST be discarded.

ARP messages where protocol is IP and sender protocol address is neither all zeros address nor bound with the corresponding binding anchor MUST be discarded.

ARP Reply messages where target protocol address is not bound with the corresponding binding anchor MUST be discarded.

For attachments with other attributes:

DHCP Server-Client messages not from attachments with the DHCP-Trust attribute or Trust attribute MUST be discarded.

For attachments with no attribute:

DHCP Server-Client messages from such attachments MUST be discarded.

The SAVI device MAY record any violation.

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 loss of attribute configuration will not break the network: no action will be performed on traffic from attachments with no attribute. However, the loss 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. The time when each binding entry is established is also saved.

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 by comparing the saved lifetime and the difference between the current time and time when the binding entry is established.

10. Constants

MAX_DHCP_RESPONSE_TIME 120s

DATA_SNOOPING_INTERVAL 60s and configurable

MAX_LEASEQUERY_DELAY 10s

OFFLINK_DELAY 30s

DETECTION_TIMEOUT 0.5s

11. Security Considerations

11.1. 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.

11.2. 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 by moving to a new attachment point or leaving the network. Since the client may return shortly, the binding should be kept, or legitimate traffic from the client will be blocked. However, if the client leaves permanently, it may be insecure to keep the binding. If 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 resources to keep bindings for departed clients.

SAVI-DHCP handles the leaving of directly attached clients. Whenever a direct client leaves, a link-down event associated with the binding anchor will be triggered. SAVI-DHCP monitors such events, and perform the following mechanism.

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

(2) If a DAD Neighbor Solicitation/Gratuitous ARP request is received that targets the address during OFFLINK_DELAY, the entry MAY be removed.

(3) If the client returns on-link during OFFLINK_DELAY, cancel the timer.

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

SAVI-DHCP does not handle the leaving of indirect clients, because it will not be notified of such events. Then the threats illustrated at the beginning of this section will be introduced. If SAVI-DHCP is enabled on indirect DHCP clients, this problem should be well understood.

11.3. Duplicate Bindings to the Same Address

The same address may be bound to multiple binding anchors only if the binding setup processes successfully complete for each binding anchor. This mechanism is designed to address the case where a client moves on the local link, and the case where a client has multiple attachments to a SAVI device.

There are two security issues with such a design:

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

Second, in the local link movement scenario, the former binding may not be removed and it can be used by an attacker sharing the same

binding anchor. For example, when a 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.

11.4. 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 by sending unicast Neighbor Solicitation/Router Solicitation/ARP Request message to determine whether a previously configured address is still valid.

Although DNA provides optimization for clients, there is insufficient information for this mechanism to migrate the previous binding or establish a new binding. If a binding is set up only by snooping the reachability test message, the binding may be invalid. For example, an attacker can perform reachability test with an address bound to another client. If binding is migrated to the attacker, the attacker can successfully 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 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 exchange at the same time, the binding will be recovered from the DHCP Snooping Process. The DHCP Request messages will not be filtered out in this case because they have link-local source addresses. Before the DHCP procedure is completed, packets will be filtered out by the SAVI device. In other words, 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 latency.

In DNAv4 [RFC4436], the ARP probe will be discarded because an unbound address is used as the 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.

11.5. 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, IPsec is generally considered heavyweight. With the deployment of this mechanism, the source IP address can be authenticated without enforcing IPsec.

By containing the DHCP servers in the protection perimeter, the DHCP servers can be protected from spoofing based attacks. Then by 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.

Again, it should be noted that although SAVI-DHCP can help authenticate the origin, it does not protect the integrity of the messages. If DHCPv6 Leasequery is performed without enforcing IPsec, this threat must be taken into account.

11.6. Binding Number Limitation

A binding entry will consume a certain high-speed memory resources. In general, a SAVI device can afford only a quite limited number of binding entries. In order to prevent an attacker from overloading the resource of the SAVI device, a binding entry limit is set on each attachment. The binding entry limit is the maximum number of bindings supported on each attachment with Validating attribute. No new binding should be set up after the limit has been reached. 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.

11.7. Privacy Considerations

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

12. 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.

13. Acknowledgment

Special thanks to Jean-Michel Combes, Christian Vogt, Joel M. Halpern, Eric Levy-Abegnoli, Marcelo Bagnulo Braun, Jari Arkko, Elwyn Davies, Barry Leiba, Ted Lemon, Leaf Yeh, Ralph Droms 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.

14. References

14.1. Normative References

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

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

[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.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

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

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

[RFC4436] Aboba, B., Carlson, J., and S. Cheshire, "Detecting Network Attachment in IPv4 (DNAv4)", RFC 4436, March 2006.

[RFC6059] Krishnan, S. and G. Daley, "Simple Procedures for Detecting Network Attachment in IPv6", RFC 6059, November 2010.

[RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", STD 37, RFC 826, November 1982.

[RFC5227] Cheshire, S., "IPv4 Address Conflict Detection", RFC 5227, July 2008.

[RFC5007] Brzozowski, J., Kinnear, K., Volz, B., and S. Zeng, "DHCPv6 Leasequery", RFC 5007, September 2007.

[RFC4388] Woundy, R. and K. Kinnear, "Dynamic Host Configuration Protocol (DHCP) Leasequery", RFC 4388, February 2006.

[I-D.ietf-opsec-dhcpv6-shield] Gont, F., Will, W., and G. Velde, "DHCPv6-Shield: Protecting Against Rogue DHCPv6 Servers", draft-ietf-opsec-dhcpv6-shield-03 (work in progress), June 2014.

[I-D.ietf-dhc-dhcpv4-over-dhcpv6] Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4 over DHCPv6 Transport", draft-ietf-dhc-dhcpv4-over-dhcpv6-09 (work in progress), June 2014.

14.2. Informative References

[RFC3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, April 2004.

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

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

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