Source Address Validation Improvement
Internet-Draft
Intended status: Standards Track
Expires: July 30, 2015

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

January 26, 2015

SAVI Solution for DHCP
draft-ietf-savi-dhcp-32

Abstract

This document specifies the procedure for creating a binding between a DHCPv4/DHCPv6-assigned IP address and a binding anchor on a Source Address Validation Improvements (SAVI) device. The bindings set up by this procedure are used to filter packets with forged source IP addresses. This mechanism complements BCP 38 ingress filtering, providing finer-grained source IP address validation.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 30, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

This document describes a fine-grained source IPv4 or IPv6 source address validation mechanism. This mechanism creates bindings between IP addresses assigned to network interfaces by DHCP and suitable binding anchors (Section 4.3.5). As discussed in Section 3 and [RFC7039], a "binding anchor" is an attribute that is immutable or difficult to change that may be used to identify the system an IP address has been assigned to; common examples include a MAC address found on an Ethernet switch port or WiFi security association. The bindings are used to identify and filter packets originated by these interfaces using forged source IP addresses. In this way, this

mechanism can prevent hosts from using IP addresses assigned to the other attachment points or invalid in the network. This behavior is referred to as "spoofing", and is key to amplification attacks, in which a set of systems send messages to another set of systems claiming to be from a third set of systems, and sending the replies to systems that don't expect them. If [RFC2827] protects a network from a neighboring network by providing prefix granularity source IP address validity, this mechanism protects a network, including a Local Area Network, from itself by providing address granularity source IP validity when DHCP/DHCPv6 is used to assign IPv4/IPv6 addresses. Both provide a certain level of traceability, in that packet drops indicate the presence of a system that is producing packets with spoofed IP addresses.

SAVI-DHCP snoops DHCP address assignments to set up bindings between IP addresses assigned by DHCP and corresponding binding anchors. It includes the DHCPv4 and v6 snooping process (Section 6), the Data Snooping process (Section 7), as well as a number of other technical details. The Data Snooping process is a data-triggered procedure that snoops the header of data packet to set up bindings. It is designed to avoid a permanent block of valid addresses in the 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, as it has nothing to do with IP address allocation. The appropriate SAVI method must be used in those cases. For hosts using Stateless Auto-Configuration to allocate addresses, 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 link-local addresses, as they are not assigned using DHCP. It is RECOMMENDED that the administration enable a SAVI solution for link-local addresses, e.g., SAVI-FCFS [RFC6620].

This mechanism works for DHCPv4-only, DHCPv6-only, or both DHCPv4 and DHCPv6. However, the DHCP address assignment mechanism in IPv4/IPv6 transition scenarios, e.g., [RFC7341], 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 physical and/or link-layer property of an attached device, as in [RFC7039]. A list of sample binding anchors can be found in Section 3.2 of that document. To the degree possible, a binding anchor associates an IP address with something unspoofable that identifies a single client system or one of its interfaces. See Section 4.3.5 for more detail.

Attribute: A configurable property of each binding anchor (port, MAC Address, or other information) that indicates the actions to be performed on packets received from the attached network device.

DHCP address: An IP address assigned via DHCP.

SAVI-DHCP: The name of this SAVI function for DHCP-assigned addresses.

SAVI device: A network device on which SAVI-DHCP is enabled.

Non-SAVI device: A network device on which SAVI-DHCP 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:

- DHCPv4 Discover: DHCPDISCOVER [RFC2131]
- DHCPv4 Request: DHCPREQUEST generated during SELECTING state [RFC2131]
- DHCPv4 Renew: DHCPREQUEST generated during RENEWING state [RFC2131]
- DHCPv4 Rebind: DHCPREQUEST generated during REBINDING state [RFC2131]
- DHCPv4 Reboot: DHCPREQUEST generated during INIT-REBOOT state [RFC2131]
- Note: DHCPv4 Request/Renew/Rebind/Reboot messages can be identified based on the Table 4 of [RFC2131]
- DHCPv4 Decline: DHCPDECLINE [RFC2131]
- DHCPv4 Release: DHCPRELEASE [RFC2131]
- DHCPv4 Inform: DHCPINFORM [RFC2131]

- DHCPv6 Request: REQUEST [RFC3315]
- DHCPv6 Solicit: SOLICIT [RFC3315]
- DHCPv6 Confirm: CONFIRM [RFC3315]
- DHCPv6 Decline: DECLINE [RFC3315]
- DHCPv6 Release: RELEASE [RFC3315]
- DHCPv6 Rebind: REBIND [RFC3315]
- DHCPv6 Renew: RENEW [RFC3315]
- DHCPv6 Information-Request: INFORMATION-REQUEST [RFC3315]

DHCP Server-to-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:

- DHCPv4 ACK: DHCPACK [RFC2131]
- DHCPv4 NAK: DHCPNAK [RFC2131]
- DHCPv4 Offer: DHCPOFFER [RFC2131]
- DHCPv6 Reply: REPLY [RFC3315]
- DHCPv6 Advertise: ADVERTISE [RFC3315]
- DHCPv6 Reconfigure: RECONFIGURE [RFC3315]

Lease time: The lease time in IPv4 [RFC2131] or the valid lifetime in IPv6 [RFC3315].

Binding entry: A rule that associates an IP address with a binding anchor.

Binding State Table (BST): The data structure that contains the binding entries.

Binding entry limit: The maximum number of binding entries that may be associated with a 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 is an access device that hosts are attached to directly. In such a case, the hosts are direct attachments (e.g., they attach directly) to the SAVI device.

Indirect attachment: A SAVI device MAY be an aggregation device that other access devices are attached to, and which hosts in turn attach to. In such a case, the hosts are indirect attachments (e.g., they attach indirectly) to the SAVI device.

Unprotected link: Unprotected links are links that connect to hosts or networks of hosts that the device may not see DHCP messages to, and are therefore outside the SAVI perimeter.

Unprotected device: An unprotected device is a device associated with an unprotected link. One example might be the gateway router of a network.

Protected link: Protected links are links that connect to hosts that the device will invariably see DHCP messages to, and are therefore within the SAVI perimeter.

Protected device: A protected device is a device associated with a protected link. One example might be a desktop switch in the network, or a host.

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 lease query exchange [RFC5007] cannot be performed by the SAVI device to fetch the lease.

4. Deployment Scenario and Configuration

4.1. Elements and Scenario

The essential elements in a SAVI-DHCP deployment scenario include a DHCP server (which may or may not be assigned an address using DHCP, and therefore may or may not be protected), zero or more protected DHCP clients, and one or more SAVI devices. It may also include DHCP relays, when the DHCP server is not co-located with a set of clients, and zero or more protected Non-SAVI devices. Outside the perimeter, via unprotected links, there may be many unprotected devices.

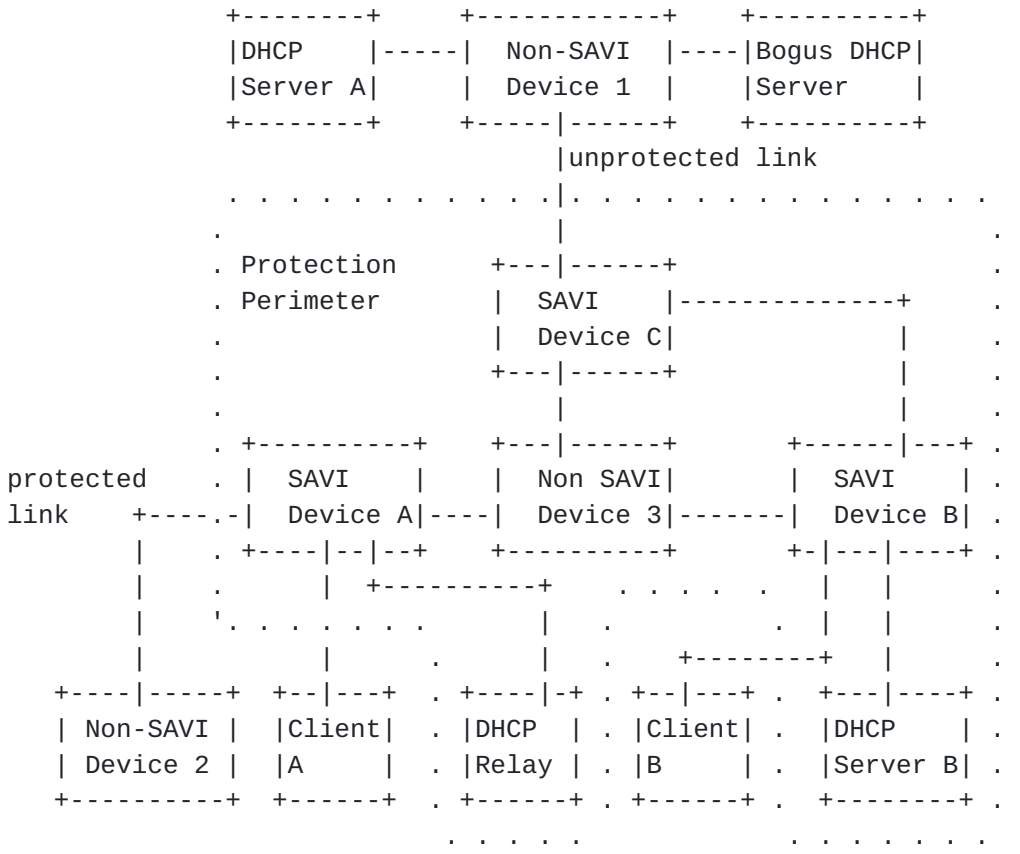

Figure 1: SAVI-DHCP Scenario

Figure 1 shows a deployment scenario that contains these elements. Note that a physical device can instantiate multiple elements, e.g., a switch can be both a SAVI device and a DHCP relay, or in a cloud computing environment, a physical host may contain a virtual switch plus some number of virtual hosts. In such cases, the links are logical links rather than physical links.

Networks are not usually isolated. As a result, traffic from other networks, including transit traffic as specified in [RFC6620] (e.g., traffic from another SAVI switch or a router) may enter a SAVI-DHCP network through the unprotected links. Since SAVI solutions are

limited to validating traffic generated from a local link, SAVI-DHCP does not set up bindings for addresses assigned in other networks and cannot validate them. Traffic from unprotected links should be checked by an unprotected system or [RFC2827] mechanisms. The generation and deployment of such a mechanism is beyond the scope of this document.

Traffic from protected links is, however, locally generated, and should be checked by SAVI-DHCP if possible. In the event that there is an intervening protected non-SAVI device between the host and the SAVI device, however, use of the physical attachment point alone as a binding anchor is insufficiently secure, as the several devices on a port or other point of attachment can spoof each other. Hence, additional information such as a MAC address SHOULD be used to disambiguate them.

4.2. SAVI Binding Type Attributes

As illustrated in Figure 1, a system attached to a SAVI device can be a DHCP client, a DHCP relay/server, a SAVI device, or a non-SAVI device. Different actions are performed on traffic originated from different elements. To distinguish among their requirements, several properties are associated with their point of attachment on the SAVI device.

When a binding association is uninstantiated, e.g., when no host is attached to the SAVI device using a given port or other binding anchor, the binding port attributes take default values unless overridden by configuration. By default, a SAVI switch does not filter DHCP messages, nor does it attempt to validate source addresses, which is to say that the binding attributes are ignored until SAVI-DHCP is itself enabled. This is because a SAVI switch that depends on DHCP cannot tell, a priori, which ports have valid DHCP servers attached, or which have routers or other equipment that would validly appear to use an arbitrary set of source addresses. When SAVI has been enabled, the attributes take effect.

4.2.1. Trust Attribute

The "Trust Attribute" is a Boolean value. If TRUE, it indicates that the packets from the corresponding attached device need not have their source addresses validated. Examples of a trusted binding anchor would be a port to another SAVI device, or to an IP router, as shown in Figure 1. In both cases, traffic using many source IP addresses will be seen. By default, the Trust attribute is FALSE, indicating that any device found on that port will seek an address using DHCP and be limited to using such addresses.

SAVI devices will not set up bindings for points of attachment with the Trust attribute set TRUE; DHCP messages and data packets from attached devices with this attribute will not be checked. If the DHCP Server-to-Client messages from attached devices with this attribute can trigger the state transitions specified in Section 6 and Section 7, the corresponding processes in Section 6 and Section 7 will handle these messages.

4.2.2. DHCP-Trust Attribute

The "DHCP-Trust Attribute" is similarly a Boolean attribute. It indicates whether the attached device is permitted to initiate DHCP Server-to-Client messages. In Figure 1, the points of attachment of the DHCP Server and the DHCP Relay would have this attribute set TRUE, and ports that are trusted would have it set TRUE.

If the DHCP-Trust Attribute is TRUE, SAVI devices will forward DHCP Server-to-Client messages from the points of attachment with this attribute. If the DHCP Server-to-Client messages can trigger the state transitions, the binding setup processes specified in Section 6 and Section 7 will handle them. By default, the DHCP-Trust attribute is FALSE, indicating that the attached system is not a DHCP server.

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

4.2.3. DHCP-Snooping Attribute

The "DHCP-Snooping Attribute" is similarly a Boolean attribute. It indicates whether bindings will be set up based on DHCP snooping.

If this attribute is TRUE, DHCP Client-Server messages to points of attachment with this attribute will trigger creation of bindings based on the DHCP snooping procedure described in Section 6. If it is FALSE, either the Trust attribute must be TRUE (so that bindings become irrelevant) or another SAVI mechanism such as SAVI-FCFS must be used on the point of attachment.

The DHCP-Snooping attribute is configured on the DHCP Client's point of attachment. This attribute can be also used on the attachments to protected Non-SAVI devices that are used 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 TRUE on a point of attachment, the "Validating Attribute" MUST also be set TRUE.

4.2.4. Data-Snooping Attribute

The "Data-Snooping Attribute" is a Boolean attribute. It indicates whether data packets from the corresponding point of attachment may trigger the binding setup procedure.

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

If the DHCP-Snooping attribute is configured on a point of attachment, the bindings on this attachment are set up based on DHCP message snooping. However, in some scenarios, a DHCP client may use a DHCP address without the DHCP address assignment procedure being performed on its current attachment. For such attached devices, 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. Since some networks require DHCP deployment and others avoid it, there is no obvious universal default value for the Data-Snooping Attribute. However, note that deployment of SLAAC (and therefore SAVI-FCFS) is generally configuration-free, while the deployment of DHCP involves at minimum the deployment of a server. Hence, the Data-Snooping Attribute should default to FALSE, and a mechanism should be implemented to conveniently set it to TRUE on all points of attachment for which the Trust attribute is FALSE.

4.2.5. Validating Attribute

The "Validating Attribute" is a Boolean attribute. It indicates whether packets from the corresponding attachment will have their IP source addresses validated based on binding entries on the attachment.

If it is TRUE, packets coming from attachments with this attribute will be checked based on binding entries on the attachment as specified in Section 8. If it is FALSE, they will not. Since the binding table is used in common with other SAVI algorithms, it merely signifies whether the check will be done, not whether it will be done for SAVI-DHCP originated bindings.

This attribute is by default the inverse of the Trust attribute; source addresses on untrusted links are validated by default. It MAY be set FALSE by the administration.

The expected use case is when SAVI is used to monitor but not block unvalidated transmissions. The network manager, in that case, may set the DHCP-Snooping and/or Data-Snooping attribute TRUE but the VALIDATING attribute FALSE.

4.2.6. Table of Mutual Exclusions

Different types of attributes may indicate mutually exclusive actions on a packet. Mutually exclusive attributes MUST NOT be set TRUE 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 to TRUE on an attachment with Trust attribute TRUE.

	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 form a perimeter separating trusted and untrusted regions of a network, as SAVI-FCFS does (Section 2.5 of [RFC6620]). The perimeter is primarily designed for scalability. It has two implications.

- SAVI devices only need to establish bindings for directly attached clients, or clients indirectly attached through a non-SAVI protected device, rather than all of the clients in the network.

- Each SAVI device only need to validate 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 Devices A, B and C. In this case, SAVI device B does not create a binding for client A. However, because SAVI device A filters spoofed traffic from client A, SAVI device B can avoid receiving spoofed 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 the DHCP Relay and DHCP Server, which are not involved in [RFC6620], is related to the construction of the perimeter. The requirement on the placement and configuration of DHCP Relay and DHCP Server are discussed in Section 4.3.3.

4.3.2. SAVI-DHCP Perimeter Configuration Guideline

A perimeter separating trusted and untrusted regions of the network is formed as follows:

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

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

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

(4) If a Non-SAVI device, or a number of connected Non-SAVI devices, are attached only to SAVI devices, set the Trust attribute TRUE on their attachments.

(5) Configure the DHCP-Trust attribute TRUE on the direct attachments to trusted DHCP relays and servers.

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

4.3.3. On the Placement of the DHCP Server and Relay

As a result of the configuration guideline, SAVI devices only trust DHCP Server-to-Client messages originated inside the perimeter. Thus, the trusted DHCP Relays and DHCP Servers must be placed within the perimeter. DHCP server-to-client messages will be filtered on the perimeter. Server-to-relay messages will not be filtered, as they are within the perimeter. In this way, DHCP server-to-client messages from bogus DHCP servers are filtered on the perimeter, having entered through untrusted points of attachment. The SAVI devices are protected from forged DHCP messages.

DHCP server-to-client messages arriving at the perimeter from outside the perimeter are not trusted. There is no distinction between a DHCP server owned and operated by the correct administration but outside the SAVI perimeter and a bogus DHCP server. For example, in Figure 1, DHCP server A is valid, but it is attached to Non-SAVI device 1. A bogus DHCP server is also attached Non-SAVI device 1. While one could imagine a scenario in which the valid one had a statistically configured port number and MAC address, and therefore a binding, by default SAVI-DHCP cannot distinguish whether a message received from the port of Non-SAVI device 1 is from DHCP server A or the bogus DHCP server. If the DHCP server A is contained in the perimeter, Non-SAVI device 1 will also be contained in the perimeter. Thus, the DHCP server A cannot be contained within the perimeter apart from manual configuration of the binding anchor.

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 common deployment practice, the traffic from the unprotected network is treated as trustworthy, which is to say that it is not filtered. In such a case, the Trust attribute can be set TRUE on the unprotected link. If Non-SAVI devices, or a number of connected Non-SAVI devices, are only attached to SAVI devices and unprotected devices, their attachment to SAVI devices can have the Trust attribute set TRUE. Then an unclosed perimeter will be formed, as illustrated in Figure 3.

To configure such a perimeter, at minimum the DHCP messages from unprotected networks MUST be ensured to be trustworthy. Achieving this is beyond the scope of this document.

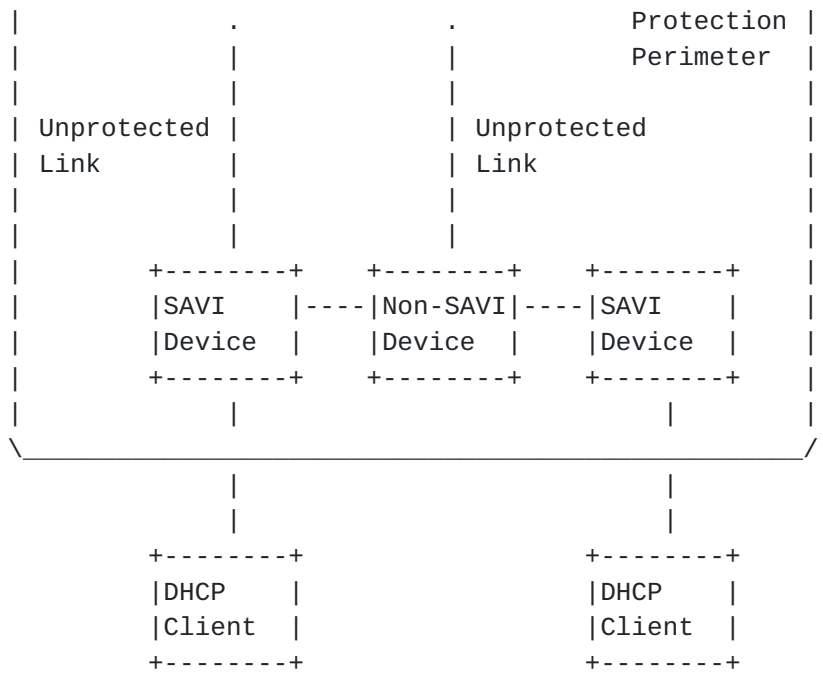

Figure 3: Alternative Perimeter Configuration

4.3.5. Considerations regarding Binding Anchors

The strength of this binding-based mechanism depends on the strength of the binding anchor. The sample binding anchors in [RFC7039] have the property that they associate an IP address with a direct physical or secure virtual interface such as a switch port, a subscriber association, or a security association. In addition, especially in the case that a protected non-SAVI device such as a desktop switch or a hub is between the client device and the SAVI switch, they MAY be extended to also include a MAC address or other link-layer attribute. In short, a binding anchor is intended to associate an IP address with something unspoofable that identifies a single client system or one of its interfaces; this may be a physical or virtual interface or that plus disambiguating link-layer information.

If the binding anchor is spoofable, such as a plain MAC address, or non-exclusive, such as a switch port extended using a non-SAVI device, an attacker can use a forged binding anchor to evade validation. Indeed, using a binding anchor that can be easily spoofed can lead to worse outcomes than allowing IP spoofing traffic. Thus, a SAVI device MUST use a non-spoofable and exclusive binding anchor.

4.4. Other Device Configuration

In addition to a possible binding anchor configuration specified in Section 4.2, an implementation has the following configuration requirements:

(1) Address configuration. For DHCPv4: the client of a SAVI device MUST have an IPv4 address. For DHCPv6: the client of a SAVI device MUST have a link-local address; when the DHCPv6 server is not on the same link as the SAVI device, the SAVI device MUST also have a global IPv6 address.

(2) DHCP server address configuration: a SAVI device MUST store the list of the DHCP server addresses that it could contact during a Lease query process.

5. Binding State Table (BST)

The Binding State Table, which may be implemented centrally in the switch or distributed among its ports, is used to contain the bindings between the IP addresses assigned to the attachments and the corresponding binding anchors of the attachments. Note that in this description, there is a binding entry for each IPv4 or IPv6 address associated with each binding anchor, and there may be several of each such address, especially if the port is extended using a protected non-SAVI device. Each binding entry, has 5 fields:

- Binding Anchor(Anchor): the binding anchor, i.e., a physical and/or link-layer property of the attachment.
- IP Address(Address): the IPv4 or IPv6 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. Internally, this MAY be stored as the timestamp value at which the lifetime expires.
- TID: the Transaction ID (TID) ([RFC2131] [RFC3315]) of the corresponding DHCP transaction. TID field is used to associate DHCP Server-to-Client messages with corresponding binding entries.

The IA is not present in the BST for three reasons:

- The lease of each address in one IA is assigned separately.

- When the binding is set up based on data-snooping, the IA cannot be recovered from the lease query protocol.

- DHCPv4 does not define an IA.

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. This process is illustrated using a state machine.

6.1. Rationale

The rationale of the DHCP Snooping Process is that if a DHCP client is legitimately using a DHCP-assigned address, the DHCP address assignment procedure that assigns the IP address to the client must have been performed on the client's point of attachment. This basis works 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 separate the DHCP client and the remaining network containing the DHCP server(s)/ and 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, if there are multiple paths from a DHCP client to the DHCP server and the SAVI device is only on one of them, there is an obvious failure case: the SAVI device may not be able to snoop the DHCP procedure. Host movement may also make this requirement difficult to meet. For example, when a DHCP client moves from one attachment to another attachment in the same network, it may fail to 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 are present in this process and the corresponding state machine:

NO_BIND: No 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-to-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-to-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 a matching binding anchor with the corresponding entry.

- TID check: the DHCP Server-to-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 Lease query and Lease query-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.

- 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 11.5).

- 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 change, and the 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 state transitions and their corresponding actions.

6.4.1. Initial State: NO_BIND - No binding has been set up

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

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

Resulting state: INIT_BIND - A potential binding has been set up

6.4.1.2. Event: EVE_DHCP_REBOOT - A DHCPv4 Reboot message is received

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.

Resulting state: INIT_BIND - A potential binding has been set up

6.4.1.3. Event: EVE_DHCP_SOLICIT_RC - A DHCPv6 Solicitation message with Rapid Commit option is received

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.

Resulting state: INIT_BIND - A potential binding has been set up

6.4.1.4. Event: EVE_DHCP_CONFIRM - A DHCPv6 Confirm message is received

The SAVI device MUST forward the message.

The SAVI device will generate corresponding entries in the BST for each address in each Identity Association (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

Resulting state: INIT_BIND - A potential binding has been set up

6.4.1.5. Events that cannot happen in the NO_BIND state

- EVE_ENTRY_EXPIRE: The lifetime of a binding entry expires
- EVE_DHCP_REBIND: A DHCPv4 Rebind or a DHCPv6 Rebind message is received

- o EVE_DHCP_RENEW: A DHCPv4 Renew or a DHCPv6 Renew message is received
- o EVE_DHCP_REPLY: A DHCPv4 ACK or a DHCPv6 Reply message is received
- o EVE_DHCP_DECLINE: A DHCPv4 Decline or a DHCPv6 Decline message is received
- o EVE_DHCP_RELEASE: A DHCPv4 Release or a DHCPv6 Release message is received
- o EVE_DHCP_LEASEQUERY: A successful DHCPv6 LEASEQUERY_REPLY is received

These cannot happen because they are each something that happens AFTER a binding has been created.

6.4.2. Initial State: INIT_BIND - A potential binding has been set up

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

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.

   A. 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.

B. Otherwise, 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 Lease query 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. 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 RECOMMENDED to use T1 configured on DHCP servers as the DHCP_DEFAULT_LEASE.)

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

Resulting state: BOUND - The binding has been set up

6.4.2.2. Event: EVE_ENTRY_EXPIRE - The lifetime of a binding entry expires

The entry MUST be deleted from BST.

Resulting state: An entry that has been deleted from the BST may be considered to be in the state "NO_BIND" - No binding has been set up.

6.4.2.3. Events that are ignored in INIT_BIND

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

As a result, the following events, should they occur, are ignored until either a DHCPv4 ACK or a DHCPv6 Reply message is received or the lifetime of the binding entry expires.

- 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_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 is received

In each case, the message MUST be forwarded.

Resulting state: INIT_BIND - A potential binding has been set up

6.4.3. Initial State: BOUND - The binding has been set up

6.4.3.1. Event: EVE_ENTRY_EXPIRE - The lifetime of a binding entry expires

The entry MUST be deleted from BST.

Resulting state: An entry that has been deleted from the BST may be considered to be in the state "NO_BIND" - No binding has been set up.

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

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.

Resulting state in each relevant BST entry: An entry that has been deleted from the BST may be considered to be in the state "NO_BIND" - No binding has been set up.

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

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.

Resulting state in each relevant BST entry: An entry that has been deleted from the BST may be considered to be in the state "NO_BIND" - No binding has been set up.

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

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.

Resulting state: BOUND: The binding has been set up

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

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.

Resulting state: BOUND: The binding has been set up

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

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 updates 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, leaving the entry in the state BOUND.

If the message is DHCPv6 Reply, the SAVI device checks each IA Address option in each IA option. For each:

1. If the IA entry in the REPLY message has the status "NoBinding", there is no address in the option, and no operation on an address is performed.

2. If the valid lifetime of an IA address option is 0, the binding entry with matched TID and address is removed, leaving it effectively in the state NO_BIND.

3. Otherwise, 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, leaving the entry in the state BOUND.

Resulting state: NO_BIND or BOUND, as specified.

6.4.3.7. Event: EVE_DHCP_LEASEQUERY - A successful DHCPv6 LEASEQUERY_REPLY is received

The message MUST be forwarded.

The message should be in response to the Lease query 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 Lease query-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.

Resulting state: BOUND: The binding has been set up

6.4.3.8. Events not processed in the state BOUND

The following events are ignored if received while the indicated entry is in the state BOUND. Any required action will be the result of the next message in the client/server exchange.

- 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_SOLICIT_RC: A DHCPv6 Solicitation message with Rapid Commit option is received

6.4.4. 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	EVE_ENTRY_EXPIRE	Remove entry	NO_BIND
BOUND	RLS/DCL	Remove entry	NO_BIND
BOUND	EVE_ENTRY_EXPIRE	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

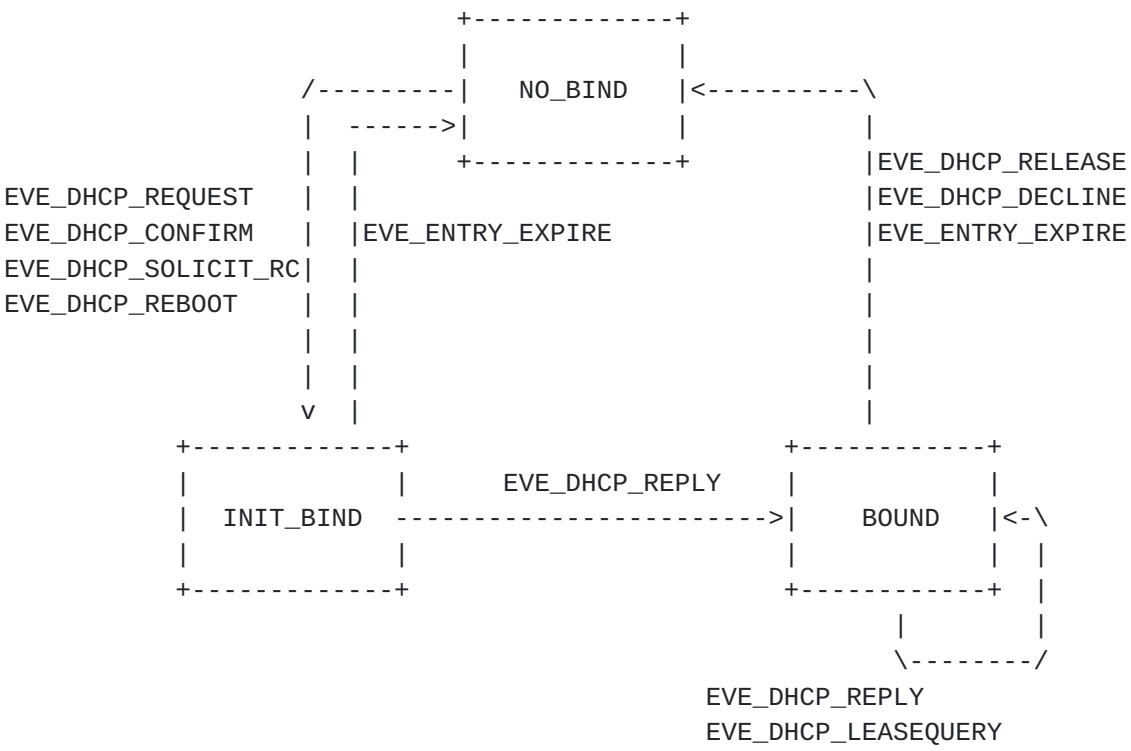

Figure 10: Diagram of Transit

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:

- Multiple paths: there is more than one feasible link-layer 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.

- Dynamic path: there is only one feasible link-layer 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 link-layer 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 NO_BIND and BOUND from Section 6.2, two new states used in this process are listed here. The INIT_BIND state is not used, as it is entered by observing a DHCP message.

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 Lease query.

7.4. Events

In addition to EVE_DATA_LEASEQUERY and EVE_DHCP_REBIND, these additional events are described here. 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 11.5).

- Address check: For EVE_DATA_LEASEQUERY, the source address of the DHCP Lease query 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].

EVE_ENTRY_EXPIRE: A timer expires.

7.5. Initial State: state NO_BIND - No binding has been set up

7.5.1. Event: EVE_DATA_UNMATCH: A data packet without matched binding is received

Make a probabilistic determination whether to act on this event. The probability may 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 the 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 an 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 Duplicate Address Detection message [RFC4861] targeting the address; ideally, only the host on that point of attachment responds with a Neighbor Advertisement; if more than one Neighbor Advertisement is observed, the BST entry should be removed.

As Duplicate Address Detection is an unreliable process (either the packet to or from the other system may be lost in transit), if there is no response, it should be repeated, as described in [RFC6620].

The packet that 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.

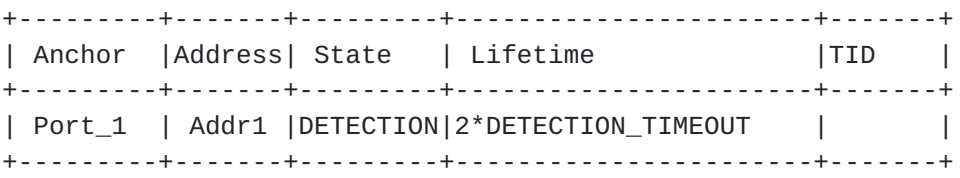

Figure 11: Binding entry in BST on data triggered initialization

Resulting state: DETECTION - The address in the entry is under local duplication detection

7.5.2. Events not observed in NO_BIND

EVE_DATA_CONFLICT: ARP Reply/Neighbor Advertisement(NA) message received from unexpected system

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

EVE_DATA_LEASEQUERY: A valid DHCPLEASEACTIVE or LEASEQUERY-REPLY is received

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

EVE_ENTRY_EXPIRE

7.6. Initial State: state DETECTION - The address in the entry is under local duplication detection

7.6.1. Event: 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.

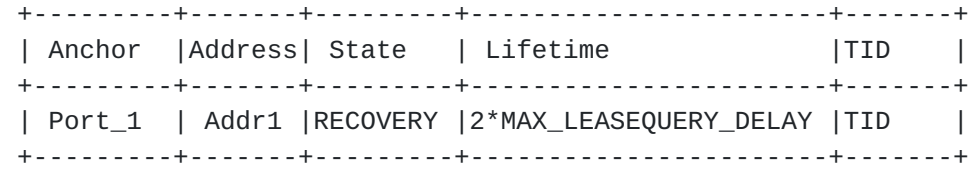

Figure 12: Binding entry in BST on Lease Query

Resulting state: RECOVERY - The SAVI device is querying the assignment and lease time of the address in the entry through DHCP Leasequery

7.6.2. Event: EVE_DATA_CONFLICT: ARP Reply/Neighbor Advertisement(NA) message received from unexpected system

Remove the entry.

Resulting state: NO_BIND - No binding has been set up

7.6.3. Events not observed in DETECTION

EVE_DATA_UNMATCH: A data packet without matched binding is received

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

EVE_DATA_LEASEQUERY: A valid DHCPLEASEACTIVE or LEASEQUERY-REPLY is received

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

7.7. Initial State: state RECOVERY - The SAVI device is querying the assignment and lease time of the address in the entry through DHCP Leasequery

7.7.1. Event: EVE_DATA_LEASEQUERY: A valid DHCPLEASEACTIVE or LEASEQUERY-REPLY is received

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, remove the entry.

(2) 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,

* 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, remove the entry.

(2) On receipt of a valid Neighbor Announcement,

* 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.

* 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.

In the event that 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.

Resulting state: if ARP or ND succeeds (there is a valid response), BOUND - The binding has been set up. Otherwise, the resulting state is NO_BIND - No binding has been set up

7.7.2. Event: EVE_ENTRY_EXPIRE

Remove the entry.

Resulting state: NO_BIND - No binding has been set up

7.7.3. Events not observed in RECOVERY

EVE_DATA_UNMATCH: A data packet without matched binding is received

EVE_DATA_CONFLICT: ARP Reply/Neighbor Advertisement(NA) message received from unexpected system

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

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

7.8. Initial State: state BOUND - The binding has been set up

Upon entry to the state BOUND, control the system continues as if a DHCP message assigning the address has been observed, as in Section 6.4.3. The BST entry has been restored.

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.

7.9. 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	EVE_ENTRY_EXPIRE	Send Leasequery	RECOVERY
DETECTION	EVE_DATA_CONFLICT	Remove entry	NO_BIND
RECOVERY	EVE_DATA_LEASEQUERY	Set lease time	BOUND or NO_BIND
RECOVERY	EVE_ENTRY_EXPIRE	Remove entry	NO_BIND
BOUND	RENEW/REBIND	Record TID	BOUND

Figure 13: Table of Transit

RENEW: EVE_DHCP_RENEW

REBIND: EVE_DHCP_REBIND

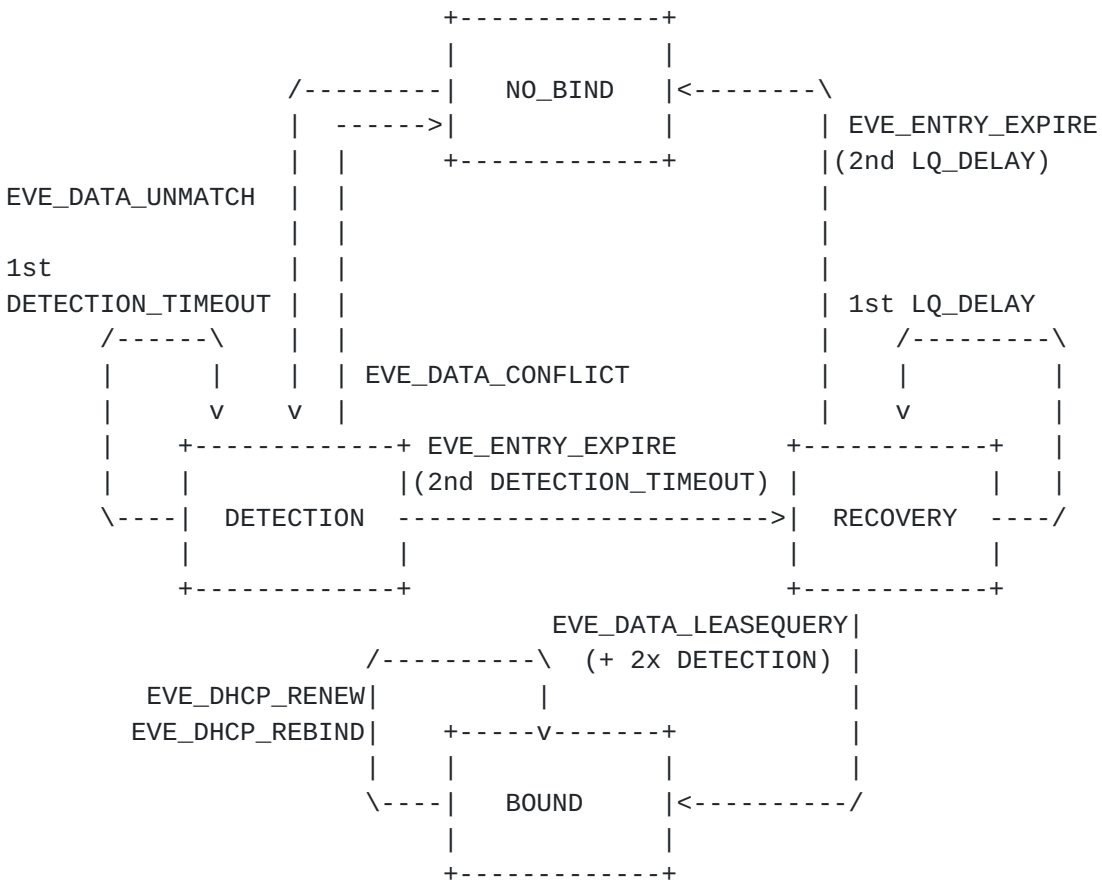

Figure 14: Diagram of Transit

LQ_DELAY: MAX_LEASEQUERY_DELAY

8. Filtering Specification

This section specifies how to use bindings to filter out packets with spoofed source addresses.

Filtering policies are different for data packets and control packets. DHCP, ARP, and NDP (Neighbor Discovery Protocol) [RFC4861] messages are classified as control packets. All other packets are classified as data packets.

8.1. Data Packet Filtering

Data packets from attachments with the Validating attribute TRUE MUST be checked. There is one exception to this rule.

A packet whose source IP address is a link-local address cannot be checked against DHCP assignments, as it is not assigned using DHCP. 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 a link-local source address.

If the source IP address of a packet is not a link-local address, but there is not a matching entry in BST with state BOUND, this packet MUST be discarded. However, the packet may trigger the Data Snooping Process Section 7 if the Data-Snooping attribute is set on the attachment.

Data packets from an attachment with the VALIDATING attribute set FALSE will be forwarded without being validated.

The SAVI device MAY log packets that fail source address validation.

8.2. Control Packet Filtering

For attachments with the Validating attribute:

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

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

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

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

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

For attachments with other attributes:

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

For attachments with no attribute:

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

The SAVI device MAY record any messages that are discarded.

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

The following constants are recommended for use in this context:

- MAX_DHCP_RESPONSE_TIME 120s (SOL_MAX_RT from [RFC3315])
- MAX_LEASEQUERY_DELAY 10s (LQ_MAX_RT from [RFC5007])
- DETECTION_TIMEOUT 0.5s (TENT_LT from [RFC6620])
- DATA_SNOOPING_INTERVAL 60s and configurable (recommendation)
- OFFLINK_DELAY 30s (recommendation)

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 be separated by a minimum interval time DATA_SNOOPING_INTERVAL. If this value is changed, the value needs to be large enough to minimize denial of service attacks.

(2) The Data Snooping Process may set up incorrect bindings if the clients do not reply to the detection probes Section 7.5.1. An

attack will pass the duplicate detection if the client assigned the target address does not reply to the detection probes. The DHCP Lease query 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. Client departure issues

After a binding is set up, the corresponding client may leave its attachment point. It may depart temporarily due to signal fade, or permanently by moving to a new attachment point or leaving the network. In the signal fade case, since the client may return shortly, the binding should be kept momentarily, lest legitimate traffic from the client be blocked. However, if the client leaves permanently, keeping the binding can be a security issue. If the binding anchor is a property of the attachment point rather than the client, e.g., the switch port but not incorporating the MAC Address, an attacker using the same binding anchor can send packets using IP addresses assigned to the client. Even if the binding anchor is a property of the client, retaining binding state for a departed client for a long time is a waste of resources.

Whenever a direct client departs from the network, a link-down event associated with the binding anchor will be triggered. SAVI-DHCP monitors such events, and performs 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 departure of indirect clients, because it will not be notified of such events. Switches supporting indirect attachment (e.g., through a separate non-SAVI switch) SHOULD use

information specific to the client such as its MAC address as part of the binding anchor.

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. 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.

11.6. 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.

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

[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.

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

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

[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.

[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.

14.2. Informative References

[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-04 (work in progress), July 2014.

[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.

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

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

[RFC7341] Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4-over-DHCPv6 (DHCP 4o6) Transport", RFC 7341, August 2014.

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