Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 25, 2014

F. Gont
SI6 Networks / UTN-FRH
R. Bonica
Juniper Networks
W. Liu
Huawei Technologies
October 22, 2013

Security Assessment of Neighbor Discovery (ND) for IPv6
draft-ietf-opsec-ipv6-nd-security-00

Abstract

Neighbor Discovery is one of the core protocols of the IPv6 suite, and provides in IPv6 similar functions to those provided in the IPv4 protocol suite by the Address Resolution Protocol (ARP) and the Internet Control Message Protocol (ICMP). Its increased flexibility implies a somewhat increased complexity, which has resulted in a number of bugs and vulnerabilities found in popular implementations. This document provides guidance in the implementation of Neighbor Discovery, and documents issues that have affected popular implementations, in the hopes that the same issues do not repeat in other implementations.

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 April 25, 2014.

Copyright Notice

Table of Contents

1. DISCLAIMER

This is WORK IN PROGRESS. Some of the recommendations might possibly change. For instance, some (NOT all) of the proposed "sanity checks" help reduce vulnerability to some attacks at the expense of e.g. reduced responsiveness. Further discussion might find some of such checks to be inadequate or inappropriate. On the other hand, some of mitigations discussed in this document have been incorporated into popular Neighbor Discovery (ND) implementations.

2. Introduction

Neighbor Discovery is used by nodes on the same link to discover each other's presence, to determine each other's link-layer addresses, to find routers, and to maintain reachability information about the paths to active neighbors [RFC4861].

Neighbor Discovery is specified by [RFC4861]. [RFC3122] specifies extensions to Neighbor Discovery for Inverse Discovery. [RFC4389] specifies Neighbor Discovery proxies. [RFC3756] describes trust models and threats for Neighbor Discovery. [RFC3971] specifies a secure version of Neighbor Discovery named 'SEcure Neighbor Discovery (SEND)'.

Neighbor Discovery was originally specified by [RFC2461], which was later obsoleted by [RFC4861]. [RFC4943] clarifies the rationale for the removal of the 'on-link assumption' from [RFC4861].

Section 3 of this document provides an analysis of each of the Neighbor Discovery messages, along with a discussion of the Neighbor Discovery options that have been specified at the time of this writing. Section 4 discusses the security implications of Router and

Prefix Discovery. Section 5 describes the security implications of Address Resolution. Section 6 contains a vulnerability analysis of Neighbor Discovery.

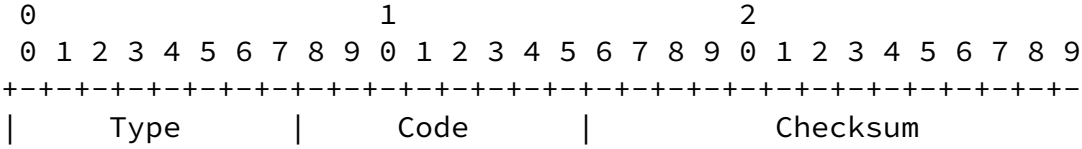
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Neighbor Discovery messages

The following subsections discuss a number of validation checks that should be performed on Neighbor Discovery messages.

3.1. Router Solicitation message

The following figure illustrates the syntax of Router Solicitation messages:



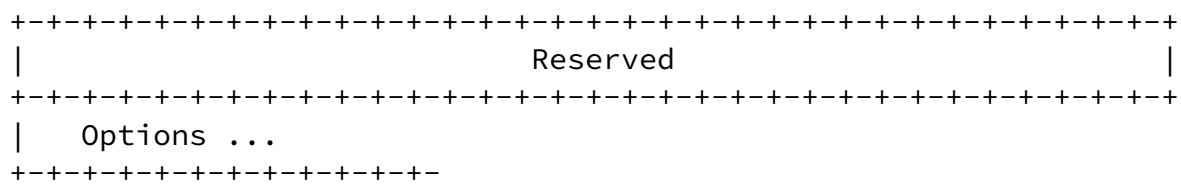


Figure 1: ICMPv6 Router Solicitation message format

As can be inferred from syntax of Router Solicitation messages, any legitimate Router Solicitation message must have a length (as derived from the IPv6 length) that is 8 octets or more. If the packet does not pass this check, it should be silently dropped.

The Source Address of an IPv6 packet encapsulating a Router Solicitation message is set to the value of one of the addresses assigned to the sending interface, or to the unspecified address (::) if no address has been assigned to that interface. Nodes should discard Router Solicitation messages that have a multicast address in the Source Address field.

The Destination Address of an IPv6 packet encapsulating a Router Solicitation message is set to the all-routers multicast address.

> A unicast address could possibly be used for the Destination Address for debugging purposes.

If a unicast address is used for the Destination Address, the receiving system should ensure that it is a link-local address. If the packet does not pass this check, it should be silently dropped.

> While this is not explicitly required in [RFC4861] this provides an additional counter-measure (other than the validation of the Hop Limit) for non-local malicious nodes willing to make use of Router Solicitation messages for reconnaissance purposes.

As of this writing, the following options are valid in a Router Solicitation message:

- o Source link-layer address

Any other options should be silently ignored.

If a 'source link-layer address' option is included, the following sanity checks should be performed:

- o The Source Address of the packet must not the unspecified address (::) or the "loopback" addresses (::1)

- o The advertised link-layer address must not a broadcast or multicast address

3.2. Router Advertisement

The following figure illustrates the syntax of Router Advertisement messages.

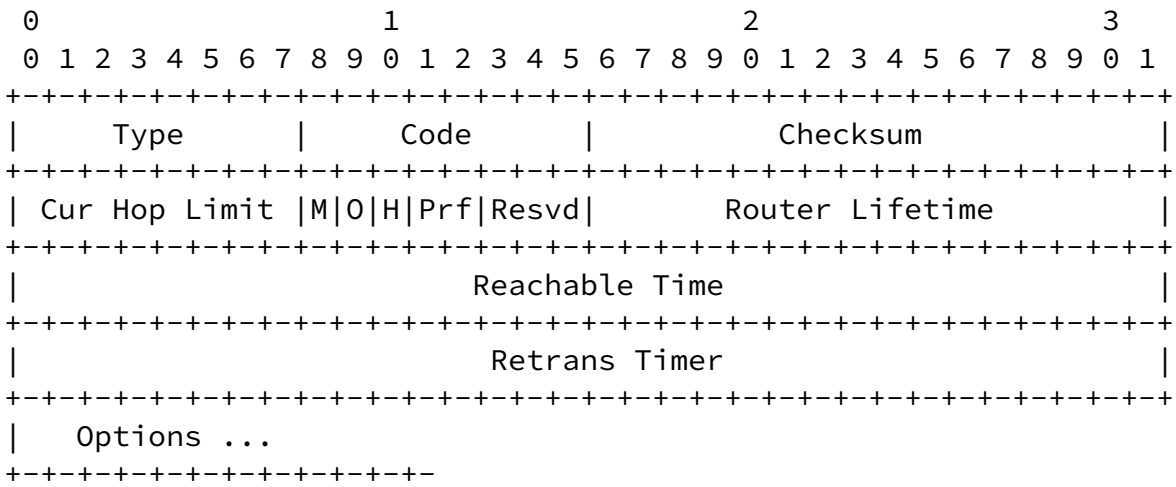


Figure 2: ICMPv6 Router Advertisement message format

The Source Address of an IPv6 packet encapsulating a Router Advertisement message is set to a link-local address assigned to the interface from which the message is sent. Nodes should discard Router Advertisements whose Source Address is not a link-local address.

The Destination Address of an IPv6 packet encapsulating a Router

Advertisement message is set to the Source Address of the system that elicited the Router Advertisement message (unless this was the unspecified address), or in the case of unsolicited Router Advertisements, to the all-nodes multicast address. Nodes receiving a Router Advertisement should ensure that if the Destination Address is a unicast address, it is a link-local address. Otherwise, the Router Advertisement message should be silently dropped.

> While this is not explicitly required in [RFC4861] this provides another mitigation for non-local malicious nodes willing to make use of Router Solicitation messages for reconnaissance purposes.

The Cur Hop Limit field specifies the default value that should be placed in the Hop Count field of outgoing IPv6 packets. As stated in [RFC4861] a value of 0 means unspecified (by this router). If the Cur Hop Limit field is larger than 0, nodes should sanitize the received Cur Hop Limit value as follows:

```
SanitizedCH = max(Cur Hop Limit, MIN_HOP_LIMIT)
```

where the sanitized Cur Hop Limit (SanitizedCH) is set to the maximum of the Cur Hop Limit and the variable MIN_HOP_LIMIT. MIN_HOP_LIMIT should default to 64, and should be configurable by the system administrator.

If the received Cur Hop Limit were not sanitized, an attacker could perform a Denial-of-Service (DoS) attack against the local network by forging a Router Advertisement message that includes a very small Cur Hop Limit value. As a result, nodes honouring the Router Advertisement would set the Hop Limit of outgoing packets to such small value, and as a result those packets would be dropped by some intervening router.

For example, if an attacker were to forge a Router Advertisement that contains a Cur Hop Limit of 1, the victim nodes could communicate only with nodes on the same network link, as their packets would be dropped by the first-hop router.

XXXX The Prf field is specified in [RFC4191] and is used to specify a 'preference' value for the router sending the Router Advertisement.

The Router Lifetime field is a 16-bit unsigned integer that specifies the lifetime associated with the default router in units of seconds. A Router Lifetime of 0 indicates that the router is not a default router and must not appear in the default router list. The sending rules in Section 6 of [RFC4861] limit the Router Lifetime to 9000 seconds. However, nodes are expected to handle any value.

An attacker could exploit the Router Lifetime field to perform DoS attacks or performance-degrading attacks. For example, an attacker could forge Router Advertisement messages that include a very small Router Lifetime. This would have a two-fold effect on the network. Firstly, once the advertised router expires as a 'default' router, the corresponding nodes might face a Denial of Service, as a result of having no default routers. Secondly, a small Router Lifetime value could lead to increased traffic in the network, and increased processing time in the affected nodes (as a result of the additional Router Solicitation/Advertisement exchanges needed to re-configure the routing table of each node).

If the Router Lifetime is different from 0, it should be sanitized as follows:

```
SanitizedRL = min( max(Router Lifetime, MIN_ROUTER_LIFETIME),
                   MAX_ROUTER_LIFETIME)
```

where lower and upper limits are enforced on the advertised Router Lifetime. The lower limit is specified by the variable MIN_ROUTER_LIFETIME, and should default to 1800 seconds. The upper limit is specified by MAX_ROUTER_LIFETIME, and should default to 9000 seconds.

The value '1800 seconds' results from the recommended default value (AdvDefaultLifetime) for setting the Router Lifetime, which instead results from the expression '3 * MaxRtrAdvInterval' (where MaxRtrAdvInterval defaults to 600 seconds). The value '9000 seconds' results from the required upper limit for setting the Router Lifetime field (AdvDefaultLifetime).

The Router Lifetime should not be sanitized when it is equal to 0, as a value of 0 indicates that the corresponding router should not be used as a default router (i.e., it is only advertising prefixes).

When a router is in the Default routers list, and a Router Advertisement is received with a Router Lifetime of 0, a node might choose to keep the router in the Default routers list (as allowed by the current local Router Lifetime value). This might allow nodes to be resilient to Router Advertisements that incorrectly or maliciously advertise a Router Lifetime of 0, at the expense of loss of responsiveness in scenarios in which a router explicitly advertises it wants to be removed from the Default routers list (such a scenario is described in Section 6.2.5 of [RFC4861].

The Reachable Time field is a 32-bit unsigned integer that specifies

the amount of time, in milliseconds, that a node assumes a neighbor is reachable after having received a reachability confirmation. A

value of zero means 'unspecified by this router'. If Reachable Time is different from 0, it should be sanitized as follows:

```
SanitizedRT = max( min( Reachable Time, MAX_REACHABLE_TIME),
                   MIN_REACHABLE_TIME)
```

where MAX_REACHABLE_TIME and MIN_REACHABLE_TIME impose upper and lower limits, respectively, to the received Reachable Time value. We propose a MAX_REACHABLE_TIME of 3,600,000 (one hour) and a MIN_REACHABLE_TIME of 20,000.

The upper limit of 3,600,000 is specified in Section 6.2.1 of [RFC4861] (AdvReachableTime router variable). The lower limit has been selected such that the minimum local ReachableTime (that would result from MIN_RANDOM_FACTOR * SanitizedRT) is not smaller than 10 seconds.

The Retrans Timer is a 32-bit unsigned integer that specifies the amount of time, in milliseconds between retransmitted Neighbor Solicitation messages. A value of zero means 'unspecified by this router'. If Retrans Timer is different from 0, it should be sanitized as follows:

```
SanitizedRXT = max( min( Retrans Timer, MAX_RETRANS_TIME),
                    MIN_RETRANS_TIME)
```

We propose a MAX_RETRANS_TIME of 60,000 and a MIN_RETRANS_TIME of 1,000.

At the time of this writing, the options that may be legitimately included in Router Advertisements are:

- Source link-layer address
- MTU
- Prefix information
- Route Information

o Recursive DNS Server

o DNS Search List

Other options should be silently ignored.

The Source link-layer address option specifies the link-layer address of the interface from which the Router Advertisement is sent. It is

only used on link layers that have addresses. Nodes should ignore the source link-layer address option in Router Advertisements received on link layers that do not have addresses.

Section 3.6 of this document discusses the security implications of all the Neighbor Discovery options.

3.3. Neighbor Solicitation message

The following figure illustrates the format of Neighbor Solicitation messages:

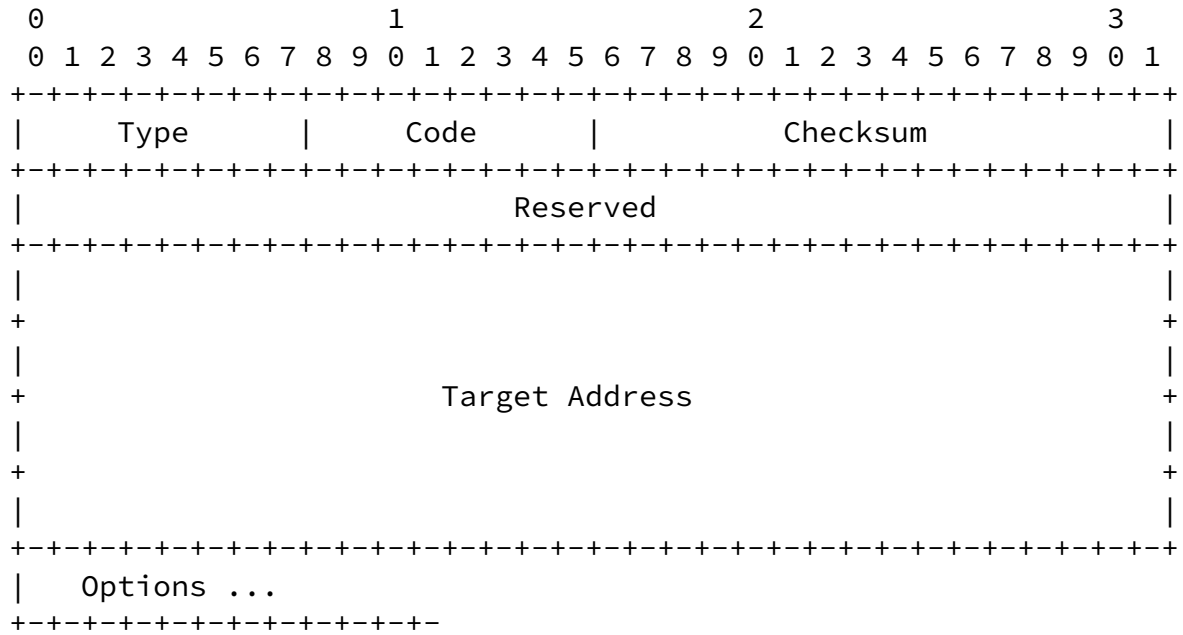


Figure 3: ICMPv6 Neighbor Solicitation message format

The Source Address of an IPv6 packet encapsulating a Neighbor Solicitation message is set to an address assigned to the interface from which the message is sent, or to the unspecified address (::).

The Destination Address of an IPv6 packet encapsulating a Neighbor Solicitation message is set to the solicited-node multicast address corresponding to the target address, or to the target address.

The ICMPv6 packet length (as derived from the IPv6 Payload Length) must be greater than or equal to 24. If the packet does not pass this check, it should be silently dropped.

The Target Address is the IPv6 address of the target of the solicitation. The Target Address must pass the following checks:

1. It must not be a multicast address (as required in Section 4.3 of [RFC4861])

2. It must not be the unspecified address (::)

3. It must not be the loopback address (::1)

The Target Address must also meet any of the following criteria:

1. It is a valid unicast or anycast address assigned to the receiving interface

2. It is a unicast or anycast address for which the node is offering proxy service

3. It is a 'tentative' address on which 'Duplicate Address Detection' (DAD) is being performed (in which case the Neighbor Solicitation message should be processed according to [RFC4862])

At the time of this writing, the options that may be legitimately included in Neighbor Solicitations are:

o Source link-layer address

According to Section 4.3 of [RFC4861], the source link-layer address

option must not be included when the Source Address is the unspecified address (::). A node receiving a Neighbor Solicitation that includes a source link-layer address and that has the unspecified address (::) as the Source Address should silently drop the corresponding packet.

According to Section 4.3 of [RFC4861], on link layers that have addresses (and provided that the Source Address is not the unspecified address), Neighbor Solicitations sent to multicast addresses must include the source link-layer address option. A node receiving a Neighbor Solicitation sent to a multicast address that does not include a source link-layer option should be silently dropped.

3.4. Neighbor Advertisement message

A node sends Neighbor Advertisements in response to Neighbor Solicitations and sends unsolicited Neighbor Advertisements in order to (unreliably) propagate new information quickly [RFC4861].

The following figure illustrates the syntax of Neighbor Advertisement messages:

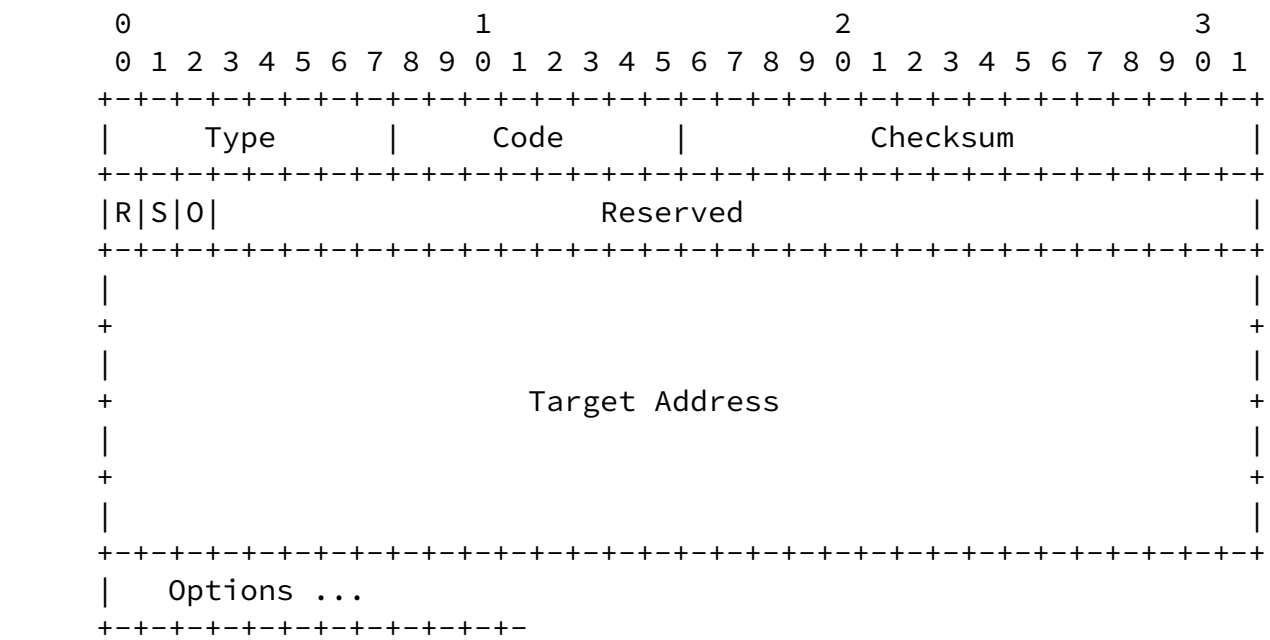


Figure 4: ICMPv6 Neighbor Advertisement message format

The Source Address of an IPv6 packet encapsulating a Neighbor Advertisement message is set to a link-local address assigned to the interface from which the message is sent. Nodes should discard Neighbor Advertisements that do not have a link-local address in the Source Address field.

The Destination Address of an IPv6 packet encapsulating a Neighbor Advertisement message is set to the Source Address of the Neighbor Solicitation that elicited the Neighbor Advertisement message (provided the Source Address of the Neighbor Solicitation was a unicast address). If the Source Address of the Neighbor Solicitation was the unspecified address, the Neighbor Advertisement is sent to the all-nodes multicast address. Finally, unsolicited Neighbor Advertisements are sent to the all-nodes multicast address

The Hop Limit of an IPv6 packet encapsulating a Neighbor Advertisement message must be set to 255 by the sending node. A node receiving a Neighbor Advertisement message should perform the following check:

The ICMPv6 packet length (as derived from the IPv6 Payload Length) must be greater than or equal to 24. If the packet does not pass this check, it should be silently dropped.

The R flag is the Router flag, and is used by Neighbor Unreachability Detection (NUD). When set, it indicates that the sender is a router. An attacker could forge a Neighbor Advertisement message with the Router flag cleared to cause the receiving node to remove the

impersonated Router from the Default router list.

The S bit is the Solicited flag. When set it indicates that the Neighbor Advertisement is sent in response to a Neighbor Solicitation sent from the Destination Address. The S flag is used as reachability confirmation for Network Unreachability Detection (NUD). As stated in Section 4.4 of [RFC4861], it must not be set in multicast advertisements or in unsolicited unicast advertisements.

A node that receives a Neighbor Advertisement message that has the S-bit set and was sent to a multicast address should silently discard

the received message. Additionally, a node that receives an unsolicited Neighbor Advertisement message (i.e., there was not a pending Neighbor Solicitation for the Target Address) with the S-bit set that was sent to a unicast address should silently drop the received message.

The O bit is the Override flag. When set, it indicates that this Neighbor Advertisement should override an existing cache entry and update the cached link-layer address. When the O bit is not set, the advertisement will not update a cached link-layer address, but will update a Neighbor Cache entry that does not include a link-layer address.

The O bit should be set in all solicited advertisements, except those for anycast addresses. A node that receives an unsolicited Neighbor Advertisement message with the O bit set should silently drop the received message. However, we note that it is virtually impossible to enforce this requirement for Neighbor Advertisement messages for anycast addresses that have the O bit set, as anycast addresses are syntactically indistinguishable from normal unicast addresses.

For solicited Neighbor Advertisements, the Target Address is set to the Target Address of the Neighbor Solicitation message that elicited the advertisement. For unsolicited Neighbor Advertisements, the Target Address is set to the address whose link-layer address has changed.

The Target Address must pass the following checks:

1. It must not be a multicast address (as required in Section 4.4 of [RFC4861]

2. It must not be the unspecified address (::)

3. It must not be the loopback address (::1)

As of this writing, the following options are allowed in Neighbor

Advertisement messages:

o Target link-layer address

Other options present in a Neighbor Advertisement should be ignored.

The target link-layer address specifies the link-layer address of the target of the Neighbor Advertisement. According to Section 4.4 of [RFC4861], this option must be included in Neighbor Advertisements when they are sent in response to neighbor solicitations sent to multicast addresses (provided the link layer has addresses). A node that receives a Neighbor Advertisement message in response to a solicitation sent to a multicast address, without a target link-layer address should silently drop the received message (provided that the corresponding link layer has addresses).

Section 3.6.3 contains further validation checks that should be performed on target link-layer address options.

3.5. Redirect message

Routers send Redirect packets to inform a host of a better first-hop node on the path to a destination, or to inform a host that the destination node is in fact a neighbor (i.e., it is attached to the same link).

The following figure illustrates the syntax of the Redirect message:

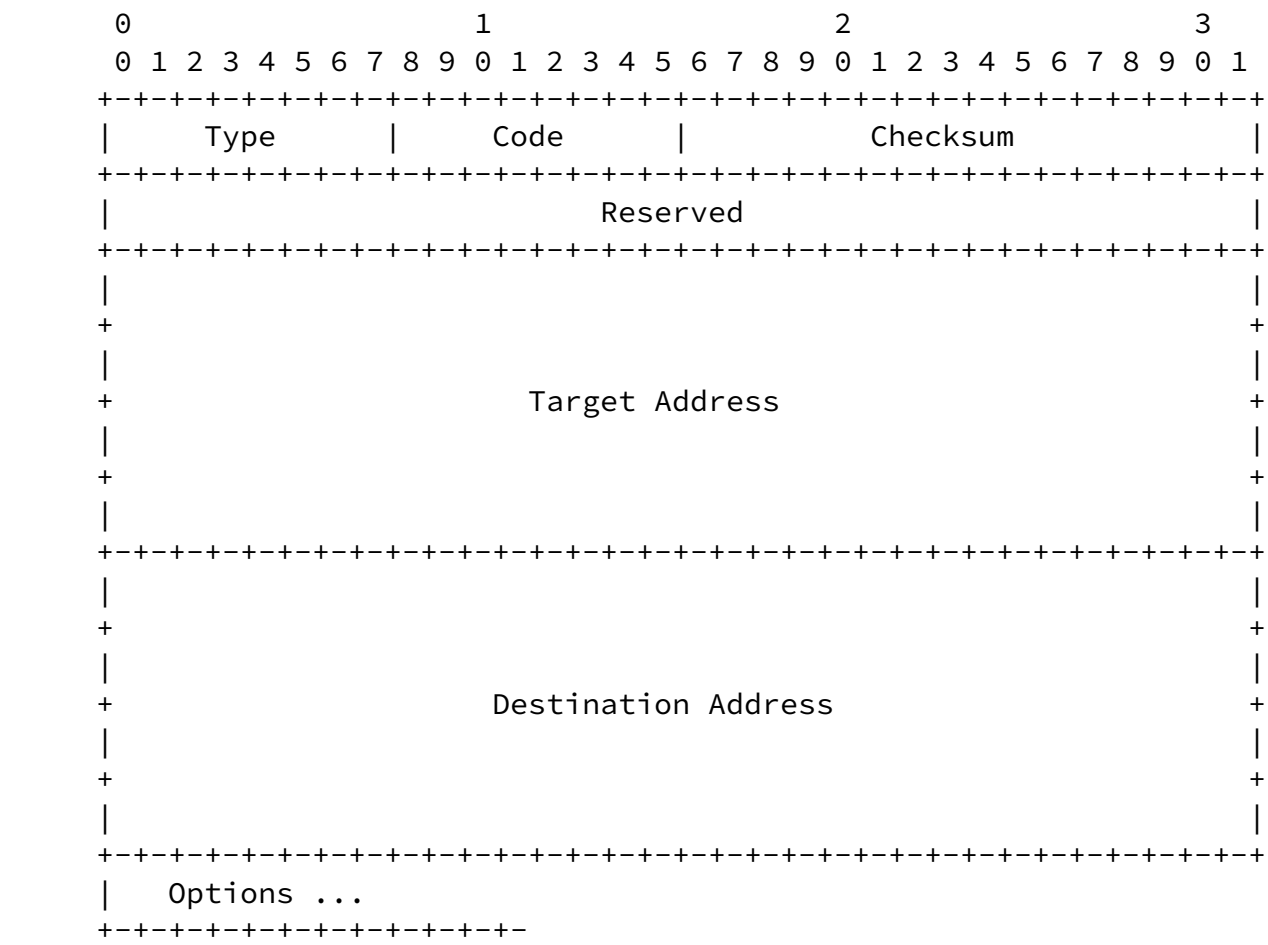


Figure 5: ICMPv6 Redirect message format

The Source Address of the IPv6 header is set to the link-local address assigned to the interface from which the Redirect message is sent. A node that receives a Redirect message should verify that the Source Address of the IPv6 header is a link-local address. If the packet does not pass this check, the Redirect message should be silently dropped. The Source Address of a Redirect message must correspond to the IPv6 address of the current first-hop router for the specified ICMPv6 Destination Address (i.e., the IPv6 address specified in the Destination Address field of the ICMPv6 Redirect message). If the packet does not pass this check, it should be silently dropped.

The Destination Address of the IPv6 header is set to the Source Address of the packet that triggered the Redirect.

The Target Address specifies an IPv6 address that is a better first hop to use for the IPv6 address specified in the Destination Address field of the ICMPv6 header. If the Redirect message is meant to indicate that a destination is in fact a neighbor (i.e., it is

attached to the same link), the Target Address is set to the same

value as the Destination Address field of the ICMPv6 header.

When the Redirect indicates a first-hop router, the Target Address must be a link-local address (that of the aforementioned 'better first-hop router'). A node that receives a Redirect message in which the Target Address and the Destination Address are different should verify that the Target Address is a link-local address. If the Redirect message does not pass this check, it should be silently dropped.

Additionally, the following checks should be performed on the ICMPv6 Target Address and the ICMPv6 Destination Address:

1. They must not contain a multicast address

2. They must not contain the unspecified address (::)

3. They must not contain the loopback address (::1)

If a Redirect message does not pass this check, it should be dropped.

As of this writing, the following options are legitimate for the Redirect message:

- Target link-layer address

- Redirected header

[RFC4861] specifies that the target-link layer address should be included (if known) in Redirect messages, and that it must be included for NBMA links that rely on the presence of the Target link-layer address option to determine the link-layer address of neighbors.

As explained in Section 8.3 of [RFC4861], if a Redirect message contains a Target link-layer address option, the node processing the redirect will create or update the Neighbor Cache entry for the target. As a result, an attacker could exploit ICMPv6 Redirect messages not only to maliciously update the Destination Cache of the victim node, but also (or alternatively) to maliciously update its

Neighbor Cache.

The Redirected header option allows the sender of the Redirect message to include a portion of the packet that triggered the Redirect message. The Redirected header option is discussed in Section 3.6.5.

3.6. Neighbor Discovery Options

Neighbor Discovery messages can include a number of options. Such options have the following general syntax:

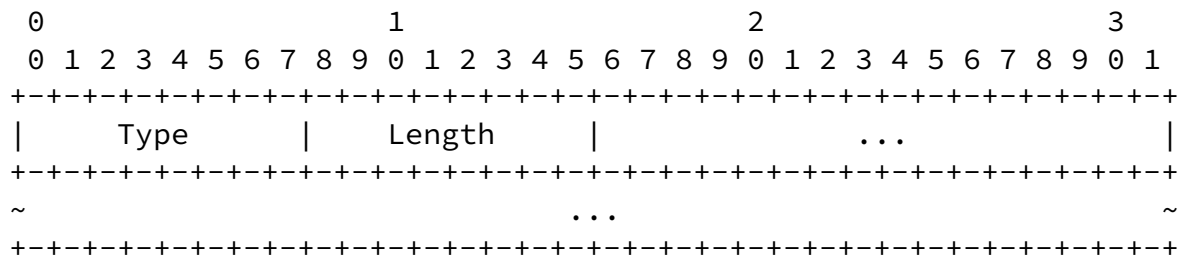


Figure 6: Neighbor Discovery option format

The Type field is an 8-bit identifier of the type of option. As of this writing, the following options have been specified:

Type	Meaning	Summary
1	Source link-layer address	Discussed in Section 3.6.2
2	Target link-layer address	Discussed in Section 3.6.3
3	Prefix information	Discussed in Section 3.6.4
4	Redirected header	Discussed in Section 3.6.5
5	MTU	Discussed in Section 3.6.6

24	Route Information	Discussed in Section 3.6.7
25	Recursive DNS Server	Discussed in Section 3.6.8
31	DNS Search List	Discussed in Section 3.6.9

Table 1: Neighbor Discovery options

The Length field specifies the length of the option in units of 8 octets. As stated in 4.6 of [RFC4861] a Length of 0 is invalid. Nodes must silently discard Neighbor Discovery packets that contain an option with a Length of 0.

3.6.1. General issues with Neighbor Discovery options

The following subsections discuss security issues that apply to all Neighbor Discovery options.

The proposed checks should be performed in addition to any option-specific checks proposed in the next sections.

Processing requirements

Processing of Neighbor Discovery options consumes CPU resources at the processing node. While the Hop Limit check of the IPv6 header encapsulating a Neighbor Discovery message limits potential attackers to those attached to the same link as the target node, there's still the potential of an on-link system overwhelming a node by sending it packets with a surprisingly large number of Neighbor Discovery options.

To reduce the impact of these packets on the system performance, a few counter-measures could be implemented:

- Rate-limit the number of Neighbor Discovery packets that are processed by the system.
- Enforce a limit on the maximum number of options to be accepted in any Neighbor Discovery message.

The first check avoids a large number of Neighbor Discovery packets to overwhelm the system in question. The second check avoids packets with multiple Neighbor Discovery options to affect the performance of the system.

Most implementations fail to rate-limit ND packets, and hence have been found vulnerable to the aforementioned issue (see e.g. [CVE-2011-2391]).

Option Length

The Length field specifies the length of the option in units of 8 octets. As stated in 4.6 of [RFC4861] a Length of 0 is invalid. Nodes must silently discard Neighbor Discovery packets that contain an option with a Length of 0. This check prevents, among other things, loops in option processing that may arise from incorrect option lengths.

Additionally, while the Length byte of a Neighbor Discovery option allows for an option length of up to 2040 octets (255 * 8 octets), there is a limit on legitimate option length imposed by the syntax of

the IPv6 header.

For all Neighbor Discovery options, the following check should be enforced:

option-offset + Length * 8 - MIN_IPV6_HEADER <= Payload Length

Where

option-offset is the offset of the first byte of the option within the IPv6 packet (with the first octet of the IPv6 header having an 'offset' of 0). Length is the Length field of the Neighbor Discovery option being processed. MIN_IPV6_HEADER is the size of the fixed IPv6 header. That is, 40 octets. Payload Length is the header field from the IPv6 header encapsulating the Neighbor Discovery message.

If a Neighbor Discovery option does not pass this check, the corresponding Neighbor Discovery message should be silently dropped.

The aforementioned check is meant to detect forged option-length values that might make an option illegitimately exceed the actual length of the IPv6 packet encapsulating the Neighbor Discovery message.

3.6.2. Source Link-Layer Address Option

The Source link-layer address option contains the link-layer address of the sender of the packet. It is used by Neighbor Solicitation, Router Solicitation, and Router Advertisement messages.

The following figure illustrates the syntax of the source link-layer address:

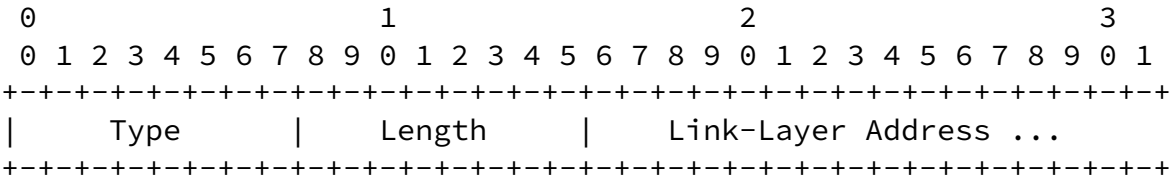


Figure 7: ND Source link-layer address option

The Type field is set to 1. The Length field specifies the length of the option (including the Type and Length octets) in units of 8 octets. A node that receives an ICMPv6 message with this option should verify that the Length field is valid for the underlying link layer. For example, for IEEE 802 addresses the Length field must be 1 [RFC2464]. If the packet does not pass this check, it should be

silently dropped.

The Link-Layer Address field contains the link-layer address. The length, contents, and format of this field varies from one link layer to another, and is specified in specific documents that describes how IPv6 operates over different link layers.

A number of validation checks should be performed on the Link-Layer Address. In the case of IEEE 802 addresses, it should not contain a broadcast or multicast address. If the option does not pass this check, the Neighbor Discovery message carrying the option should be discarded.

Additionally, nodes should not allow the source link-layer address to contain one of the receiving node's link-layer addresses. If the option does not pass this check, the Neighbor Discovery message carrying the option should be discarded.

The source link-layer address option could be exploited for the purpose of 'Neighbor Cache poisoning', that is, to cause traffic meant for a specific IPv6 address to be illegitimately directed to the node whose link-layer address is specified by the Link-Layer Address field.

This is similar to the ARP cache poisoning attacks in IPv4.

A possible counter-measure for Neighbor Cache poisoning attacks would be to override the link-layer address stored in the Neighbor Cache only after Neighbor Unreachability Detection (NUD) finds the neighbor to be unreachable and the corresponding entry is removed. This is clearly a trade-off between responsiveness and resiliency.

In some network scenarios it may be possible and desirable to configure static Neighbor Cache entries, such that Neighbor Discovery need not be performed for the corresponding IPv6 addresses.

Some implementations have been found to inadvertently override static entries when they receive source link-layer address options or target link-layer address options in Neighbor Discovery messages [Hogg-Vyncke] [Lecigne-Neville-Neil].

If source link-layer address options were allowed to contain broadcast (e.g., the IEEE 802 'ff:ff:ff:ff:ff:ff' address) or multicast (e.g., the IEEE 802 '33:33:00:00:00:01' address) addresses, traffic directed to the corresponding IPv6 address would be sent to the broadcast or multicast address specified in the source link-layer option. This could have multiple implications:

It would have a negative impact on the performance of the nodes attached to the network and on the network itself, as packets sent to these addresses would need to be delivered to multiple nodes (and processed by them) unnecessarily.

An attacker could capture network traffic sent to the corresponding IPv6 address, as the corresponding packets would be delivered to all (in the case of broadcast) or multiple (in the case of multicast) nodes.

Packets could result in forwarding loops at routers, as a router forwarding a packet to the corresponding address would receive itself a copy of the forwarded packet, thus resulting in a forwarding loop. The loop would end only when the Hop Limit is eventually decremented to 0. If multiple routers are present on the same link, the problem is further exacerbated. Section 6.1.10 of this document contains further analysis of this vulnerability.

[Lecigne-Neville-Neil] reports that at least some versions of FreeBSD are vulnerable to these issues.

3.6.3. Target Link-Layer Address Option

The Target link-layer address option contains the link-layer address of the Target of the packet. It is used by Neighbor Advertisement and Redirect messages.

The following figure illustrates the syntax of the Target link-layer address:

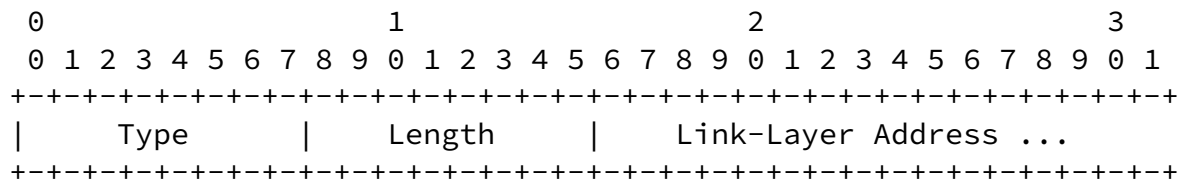


Figure 8: ND Target link-layer address option format

The Type field is set to 2. The Length field specifies the length of the option (including the Type and Length octets) in units of 8 octets. A node that receives an ICMPv6 message with this option should verify that the Length field is valid for the underlying link-layer. For example, for IEEE 802 addresses the Length field must be 1 [RFC2464]. If the packet does not pass this check, it should be silently dropped.

The Link-Layer Address field contains the link-layer address. The

length, contents, and format of this field varies from one link layer to another, and is specified in specific documents that describes how IPv6 operates over different link layers.

The target link-layer address has the same security implications as the source link-layer address. Therefore, the same considerations apply, and the same validation checks should be performed as for the source link-layer address (see Section 3.6.2).

3.6.4. Prefix Information

The Prefix Information option is used by routers to provide hosts with on-link prefixes and prefixes for Address Auto-configuration. It may only appear in Router Advertisement messages and should be silently ignored in any other messages [RFC4861].

The following figure illustrates the syntax of the Prefix Information option:

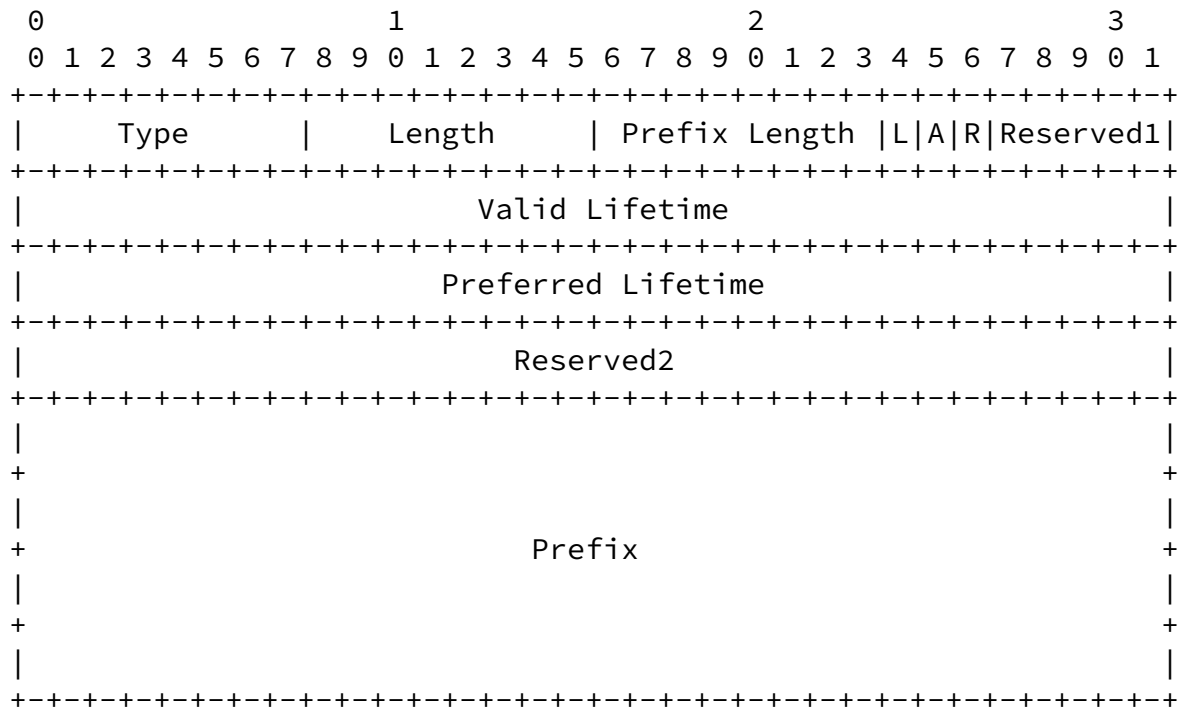


Figure 9: ND Prefix Information option format

The Type field is set to 3. The Length field is set to 4 by the sender. A node processing a Prefix Information option should verify that the Length field is 4. If the option does not pass this check, the option should be ignored.

The Prefix Length is an 8-bit unsigned integer that specifies the

prefix length, that is, the number of leading bits in the Prefix field that are valid.

The following sanity check should be applied on the Prefix Length field:

Prefix Length >= 32

If the Prefix Length field does not pass this checks, the Prefix Information option should be discarded.

The L bit is a 1-bit flag that, when set, states that the prefix can be used for on-link determination. The A bit is a 1-bit autonomous address-configuration flag that indicates whether this prefix can be used for autonomous address configuration. The R flag is specified by [RFC6275], and indicates that the Prefix field contains a complete IPv6 address assigned to the sending router. The Reserved1 field is a 6-bit unused field that is set to zero by the sender and must be ignored by the receiver.

The Valid Lifetime field is a 32-bit unsigned integer that specifies the amount of time (in seconds) this prefix can be used for on-link determination (with a value of 0xffffffff representing 'infinity'). We recommend hosts to sanitize the Valid Lifetime as follows:

SanitizedVL = max(Valid Lifetime, MIN_VALID_LIFETIME)

Where SanitizedVL is the sanitized 'Valid Lifetime', and MIN_VALID_LIFETIME is set to 1800 (seconds).

The value of 1800 seconds for MIN_VALID_LIFETIME has been selected to coincide with the lower limit enforced on the Router Lifetime (MIN_ROUTER_LIFETIME).

The Preferred Lifetime is a 32-bit unsigned integer that specifies the length of time (in seconds) that addresses generated from this prefix via stateless address auto-configuration (SLAAC) should remain 'preferred' (with a value of 0xffffffff representing 'infinity').

As noted in [RFC4861] the Preferred Lifetime must be smaller than or equal to the Valid Lifetime to avoid preferring addresses that are no longer valid. Therefore, a node processing a Prefix Information

option should perform the following check:

Preferred Lifetime <= Valid Lifetime

If the option does not pass this check, it should be silently ignored.

The Reserved2 is a 32-bit unused field that is set to zero by the sender and must be ignored by the receiver.

The Prefix field contains an IPv6 address or a prefix of an IPv6 address.

The Prefix Length contains the number of leading bits in the prefix that are to be considered valid. The remaining bits in the Prefix field are set to zero by the sender and must be ignored by the receiver.

As stated in Section 4.6.2 of [RFC4861], routers should not send a Prefix Information option for the link-local prefix. Therefore, a node should verify that the Prefix does not contain the link-local prefix. If the option does not pass this check, it should be silently dropped.

Additionally, a node should verify that the Prefix does not contain a multicast IPv6 prefix. If the option does not pass this check, it should be silently dropped.

An attacker could exploit the Prefix information option to perform a Denial-of-Service attack, by sending a large number of Router Advertisements with the Prefix Information options that have the A bit set, therefore advising the receiving systems to configure an IPv6 address with each of these prefixes. If an implementation does not enforce a limit on the number of addresses they configure in response to Router Advertisements, the aforementioned attack might result in buffer overflows or kernel memory exhaustion.

[CVE-2010-4669] is one vulnerability report about the aforementioned issue.

We recommend hosts to default to a maximum number of configured addresses (for each interface) of 16.

This limits is already being enforced by a number of implementations, such as OpenBSD 4.2.

On the other hand, Windows XP SP2 and FreeBSD 9.0 fail to enforce limits on the maximum number of configured addresses, and therefore are vulnerable to a Denial of Service attack.

Even if hosts do enforce a limit on the number of IPv6 addresses configured, an attacker might try to cause victim hosts to ignore legitimate prefixes previously advertised for address configuration by legitimate routers. Hereby we recommend hosts to not discard previously configured addresses if new prefixes for address auto-

configuration are advertised and the limit for the maximum number of configured addresses (per interface) has been reached. When such limit is hit, the newly advertised prefixes for address auto-configuration should be ignored.

Section 3.6.4 describes how an attacker could exploit the Prefix Information option for the purpose of traffic hijacking.

3.6.5. Redirected Header Option

The Redirected Header option is used in Redirect messages to convey all or part of the packet that is being redirected. It must be silently ignored for all other messages.

The following figure illustrates the syntax of the Redirected Header option:

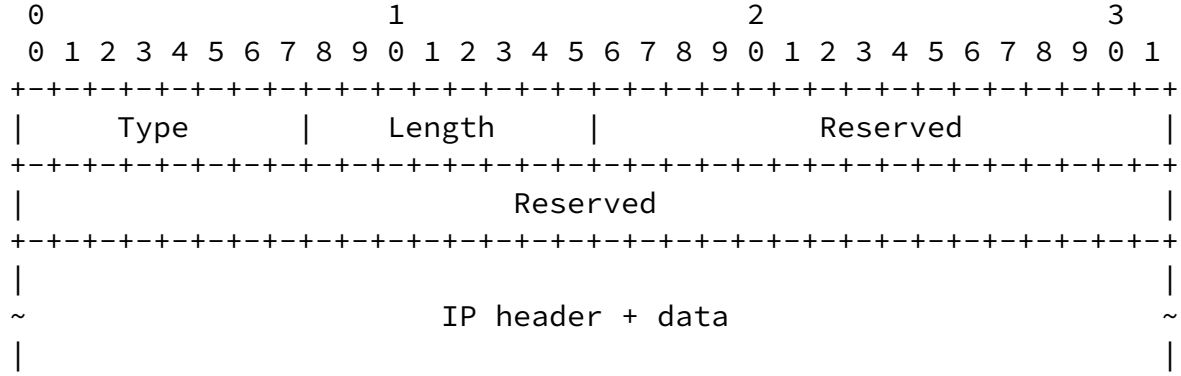


Figure 10: ND Redirected Header option format

The Type field is 4. The Length field specifies the option size (including the Type and Length fields) in units of 8 octets. Assuming the Redirected Header option will contain at least the mandatory fields of the option (8 bytes), the fixed IPv6 header (40 bytes), and the first 8 bytes of the transport protocol header, the following validation check should be performed:

Length >= 7

If the option does not pass this check, the corresponding Redirect Message should be silently ignored.

As the option is meant to contain as much of the Redirected packet without exceeding the minimum IPv6 MTU, and the minimum IPv6 MTU is 1280 octets, this is a sensible requirement to enforce.

3.6.6. MTU Option

The MTU option is used in Router Advertisement messages to ensure that all nodes on a link use the same MTU value in those scenarios in which heterogeneous technologies are bridged together. This option must be silently ignored for other Neighbor Discovery messages.

The following figure illustrates the syntax of the MTU option:

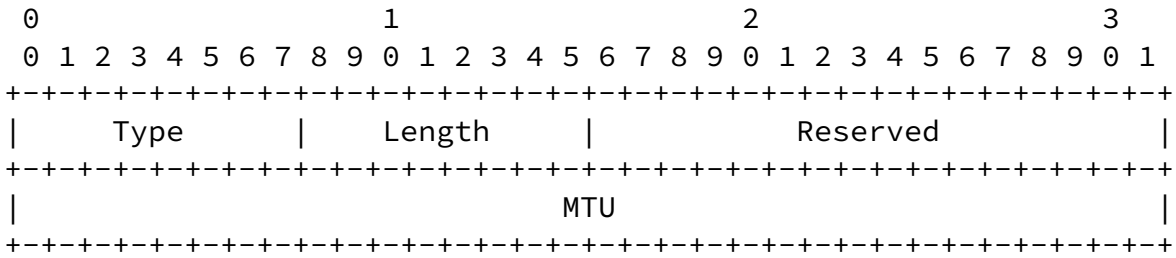


Figure 11: ND MTU option format

The Type field identifies the kind of option and is set to 5. This

option has a fixed Length that is 1. Therefore, the following sanity check should be performed:

Length == 1

If the option does not pass this check it, should be ignored.

The Reserved field is a 16-bit unused field that is set to 0 by the sender and should be ignored by the receiver.

The MTU field is a 32-bit unsigned integer that specifies the MTU value that should be used for this link. [RFC2460] specifies that the minimum IPv6 MTU is 1280 octets. Therefore, a node processing a MTU option should perform the following check:

MTU >= MINIMUM_IPV6_MTU

where MINIMUM_IPV6_MTU is a variable that should be set to 1280.

If the option does not pass this check, it should be silently ignored.

It has been reported that some link layers do not support a minimum MTU of 1280 bytes. Therefore, implementations should provide the means to change the default value of the MINIMUM_IPV6_MTU variable.

Additionally, the advertised MTU should not exceed the maximum MTU

specified in the link-type-specific document (e.g., [RFC2464] for Ethernet networks). Therefore, a node processing a MTU option should perform the following check:

MTU <= MAX_LINK_MTU

where MAX_LINK_MTU is a variable that should be set according to the maximum link MTU specified in the link-type-specific document (e.g., [RFC2464] for Ethernet).

If the option does not pass this check, it should be silently ignored.

The MTU option could be potentially exploited by an attacker to

perform a Denial-of-Service (DoS) or a performance-degrading attack against the systems in a local network. In order to perform this attack, an attacker would forge a Router Advertisement that includes an MTU option with a very small (possibly zero) value. The impact of this attack would be the same as the 'Blind performance-degrading attack' described in Section 15.7 of [CPNI-TCP].

3.6.7. Route Information Option

The Route Information option is used to convey more-specific routes in Router Advertisement messages. The following figure illustrates the syntax of the Router Information option:

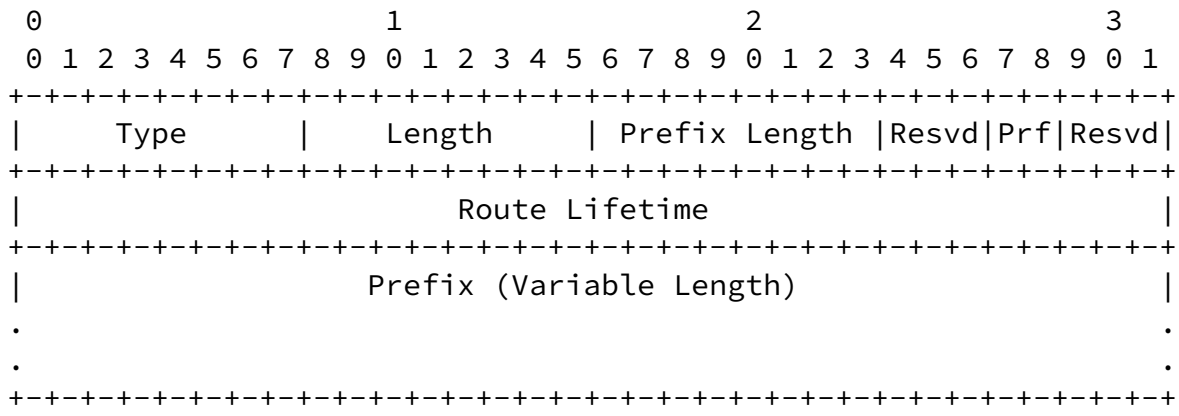


Figure 12: ND Route Information option format

The Type field identifies the type of option and is set to 24. The Length field contains the length of the option (including the Type and Length fields) in units of 8 octets. The following sanity checks should be performed on these Length field:

(Length >=1) && (Length <=3)

If the option does not pass this check, it should be ignored.

An option Length of 1 octet allows the specification of prefixes with a length of 0 (i.e., /0), while an option Length of 3 allows the specification of prefixes of up to 128 bits (i.e., /128).

The Prefix Length field indicates the number of leading bits in the

Prefix field that are valid. The Length field and the Prefix Length field are closely related, as the Length field constrains the possible values of the Prefix Length field.

The following sanity check should be enforced on the Prefix Length field:

```
Prefix Length <= (Length - 1) * 64
```

If the option does not pass this check, it should be ignored.

Both of the Rsvd fields are set to zero by the sender of the message, and should be ignored by the receiver. The Prf field specifies the 'Preference' of this route. As specified by RFC 4191, if the Prf field contains the value of '10' ('Reserved'), the option should be ignored.

The Route Lifetime field specifies the length of time, in seconds, that the prefix is valid for route determination. While not required by RFC 4191, we recommend hosts to perform the following sanity check on the Route Lifetime field:

```
SanitizedRL = min( max(Route Lifetime, MIN_ROUTE_LIFETIME),
                   MAX_ROUTE_LIFETIME)
```

Where MIN_ROUTE_LIFETIME is set to 1800 seconds and MAX_ROUTE_LIFETIME is set to 9000 seconds.

These values have been selected according to the upper and lower limits described in Section 3.2 ('Router Advertisement') of this document for the Router Lifetime field of Router Advertisements.

The Prefix field contains the actual IPv6 prefix that, together with the Prefix Length field, identifies the route. A number of sanity checks should be enforced on the Prefix field. For example, the Prefix should neither contain a link-local unicast prefix (e.g., fe80::/10, fe80::/64, etc.) nor a link-local multicast prefix (e.g., ff02::0/64).

The Route information option has a number of security implications. Firstly, an attacker could forge Router Advertisements with a higher

'preference' value (Prf), thus overriding the existing default routers at the attacked system. Secondly, an attacker could exploit this option to implant more specific routes to a victim prefix at the attacked system, thus overriding the existing routes for that victim prefix. Thirdly, an attacker could cause an existing route to a victim prefix to be removed from the routing table of the attacked host, by forging a Route Information option that contains a Route Lifetime of 0 (or some other small value). Fourthly, if an implementation does not enforce limits on the size of the Destination Cache, an attacker could possibly exhaust the kernel memory or CPU cycles of that system by forging a large number of Route Information options (possibly in many different Router Advertisements), such that the attacked system consumes its kernel memory and/or its CPU time to install those routes (see e.g. [CVE-2012-notyet]).

A general mitigation for the security implications of Router Advertisements that can be applied when the protocols are deployed is to restrict which ports of a managed switch can send Router Advertisement messages. That is, Router Advertisements received on all other ports of the switch would be discarded. This mechanism is commonly-known as Router Advertisement Guard (RA-Guard) [RFC6104] [RFC6105] [I-D.ietf-v6ops-ra-guard-implementation].

We recommend hosts to not simply discard a default router entry when a Router Preference with a higher Prf value is received. In particular, default routers that are known to be working should not be discarded when such Router Advertisements are received.

> This means that the higher-priority router would override the existing default router, but the latter would still be kept in the "default routers list", such that if the newly-learned router is found to be non-working, the existing (lower-priority) router could still be employed).

We recommend hosts to enforce a lower limit Prefix Length in the Route Information options. This would prevent an attacker from overriding the default routers by including, e.g., one Route Information option for the prefix ::/1 and one Route Information option for the prefix 8000::/1. We recommend hosts to enforce a minimum Prefix Length of 32. Hosts may also enforce an upper limit on the Prefix Length, such that an attacker cannot easily redirect traffic to specific site. A possible upper limit for the Prefix Length would be 64. As discussed earlier in this Section, the Route Lifetime value should be sanitized, and a route entry should not simply be discarded when a Route Information option with a Route Lifetime of 0 is received.

Finally, hosts should enforce a limit on the maximum number of

entries in the Destination Cache.

3.6.8. Recursive DNS Server Option

The Recursive DNS Server (RDNSS) option provides a mechanism for routers to advertise the IPv6 addresses of one or more Recursive DNS Servers. This option is specified in [RFC6106]. The following figure illustrates the syntax of the RDNSS options:

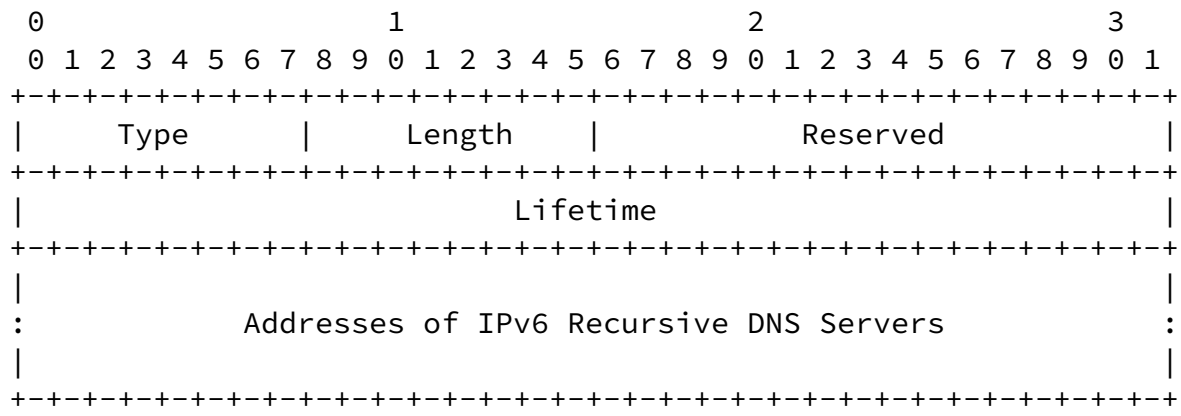


Figure 13: ND Recursive DNS Server option format

The Type field identifies must be 25. The Length field specifies the length of the option (including the Type and Length fields) in units of 8 octets. The following sanity checks should be performed on the Length field:

Length >= 3

If the option does not pass this check, it should be ignored.

This sanity check requires a RDNSS option to contain the IPv6 address of at least one Recursive DNS Server.

Additionally, the following sanity check should be performed:

(Length -1) % 2 == 0

If the option does not pass this check, it should be silently ignored.

As an IPv6 address consists of 16 bytes, each IPv6 address that is included in the option should increment the minimum option length by 2.

The Reserved field is set to zero by the sender, and must be ignored

by the receiver. The Lifetime field specifies the maximum time in seconds that a node may use the IPv6 addresses included in the option for name resolution, with a value of 0 indicating that they can no longer be used. As specified in [RFC6106], the following sanity checks should be performed on the Lifetime field:

Lifetime >= 1800

Lifetime <= 3600

[RFC6106] specifies these sanity checks as MaxRtrAdvInterval <= Lifetime <= 2*MaxRtrAdvInterval.

If the Lifetime field does not pass this check, the option should be ignored.

Failure to enforce a lower limit on the Lifetime value could allow an attacker to 'disable' a Recursive DNS Server at a target system, by forging a Router Advertisement with a RDNSS option that includes the IPv6 address of such DNS Server, and a Lifetime of 0 (or some other small value). This would cause the receiving system to remove such RDNSS address from the list of Recursive DNS Servers. However, it should be noted that this represents a trade-off of responsiveness vs. resiliency.

Sanity checks should be performed on the IPv6 addresses that are included in the RDNSS option. For example, nodes should check that the IPv6 addresses included in the RDNSS option are not multicast addresses. If any of the addresses in the RDNSS option does not pass this check, it should be silently dropped.

Nodes should enforce a limit on the number of IPv6 addresses they include in the local list of Recursive DNS Servers. An arbitrary limit could be to allow a maximum of 16 IPv6 addresses in the list of Recursive DNS Servers.

The value 16 is somewhat arbitrary. It has been chosen to be the same as the limit on the maximum number of default routes that many systems (such as OpenBSD 4.2) already enforce.

Failure to enforce limits on the maximum number of Recursive DNS Servers could probably allow an attacker to exhaust the system memory by crafting multiple Router Advertisements that advertise a large number of IPv6 addresses in RDNSS options.

It should also be clear that if an attacker is able to advertise a malicious Recursive DNS Server to victim nodes, he could perform a

variety of attacks against the victim nodes (DoS, Man-in-the-Middle. Etc.).

3.6.9. DNS Search List

The Recursive DNS Server (RDNSS) option provides a mechanism for routers to advertise the IPv6 addresses of one or more Recursive DNS Servers. This option is specified in [RFC6106]. The following figure illustrates the syntax of the RDNSS options:

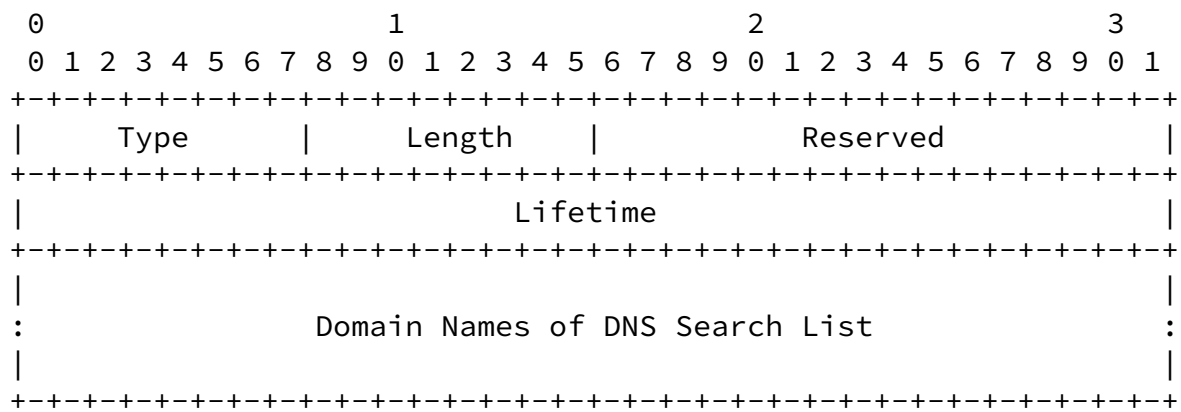


Figure 14: V

XXX (PLACEHOLDER): Need to complete the security assessment of this option.

4. Router and Prefix Discovery

4.1. Router Specification

Section 6.2 of [RFC4861] contains the Router specification for Router and Prefix Discovery.

Section 6.2.6 ('Processing Router Solicitations') of [RFC4861] states that if a router receives a Router Solicitation message, and 'the router already has a Neighbor Cache entry for the solicitation's sender, the solicitation contains a Source Link-Layer Address option, and the received link-layer address differs from that already in the cache, then the link-layer address SHOULD be updated in the appropriate Neighbor Cache entry'. As a result, an attacker might forge a Router solicitation message with a forged source link-layer address, thus causing all traffic meant from the attacked router to the (forged) Source Address of the Router Advertisement to be sent to the link-layer address advertised in the forged source link-layer address option.

Section 6.2.6 of [RFC4861] further states that 'Whether or not a Source Link-Layer Address option is provided, if a Neighbor Cache

entry for the solicitation's sender exists (or is created) the entry's IsRouter flag MUST be set to FALSE'. As a result, in a network scenario in which there are two routers ('A' and 'B') on the same link, and an attacker is directly attached to that link, an attacker could send a Router Solicitation to one of the routers (Router A) forging the Source Address to be that of the other router (Router B). As a result, the target router (Router A) would set the IsRouter flag of the Neighbor Cache entry corresponding to the IPv6 address of Router B (the forged Source Address of the Router Solicitation message) to FALSE, and as a result, Router B would be eliminated from the Default router list of Router A.

One interesting aspect about this attack is that while some devices might be filtering e.g. Router Advertisements, they might not be filtering Router Solicitation messages, and thus this attack might still be effective.

4.2. Host Specification

Section 6.3.4 of [RFC4861] states that when a Router Advertisement is received that communicates information for a specific parameter (e.g., link MTU) that differs from information received in previous Router Advertisements, the most recently received information is considered authoritative.

While this requirement guarantees that the relevant information can

be updated in a timely fashion, it also guarantees that an attacker connected to the local link always has the chance to maliciously override the values of parameters previously learned from Router Advertisements.

Section 6.3.4 of [RFC4861] states that 'to limit the storage needed for the Default Router List, a host MAY choose not to store all of the router addresses discovered via advertisements'. Here we strongly advise hosts to enforce a limit on the maximum number of entries in the Default Router List. A possible (somewhat arbitrary) limit for the maximum number of entries in the Default Router list would be 16.

Section 6.3.4 of [RFC4861] states that 'If the received Cur Hop Limit value is non-zero, the host SHOULD set its CurHopLimit variable to

the received value'. Here we strongly advise that the received Cur Hop Limit is sanitized as described in Section 3.2 of this document.

Section 6.3.4 of [RFC4861] states that 'The RetransTimer variable SHOULD be copied from the Retrans Timer field, if the received value is non-zero'. Here we strongly advise that the received Retrans Timer is sanitized as described in Section 3.2 of this document.

Honouring very small Retrans Timer values could lead the system to flood the network with Neighbor Advertisements.

With respect to the processing of received MTU options, Section 6.3.4 of [RFC4861] correctly states that the received option should be honoured as long as the received value is within the expected limits. Section 3.6.6 of this document discusses a number of checks that should be performed on received MTU options.

Section 6.3.4 of [RFC4861] states that 'The only way to cancel a previous on-link indication is to advertise that prefix with the L-bit set and the Lifetime set to zero'. This means that if an attacker forges a Router Advertisement that advertises a 'victim' prefix as being on-link, such prefix will usually be considered 'on-link' for the advertised Lifetime period of time ('forever' if Lifetime was set to 0xffffffff).

5. Address Resolution

In the case of broadcast link-layer technologies, in order for a system to transfer an IPv6 datagram, it must first map an IPv6 address to the corresponding link-layer address via Neighbor Solicitation/Advertisement messages.

While this operation is being performed, the packets that require

such a mapping would need to be kept in memory. This may happen both in the case of hosts and in the case of routers.

The possible implementation approach of keeping those datagrams in memory while the mapping operation is being performed is mentioned in Section 5.2 of [RFC4861].

This situation might be exploited by an attacker to perform a Denial-of-Service (DoS) attack against an IPv6 router, by sending a large number of packets to a non-existent node that would supposedly be a neighbor to the attacked router. While trying to map the corresponding IPv6 address into a link-layer address, the attacked router would keep in memory all the packets that would depend on that address resolution operation in order to be forwarded to the intended destination. At the point in which the mapping function times out, the node would typically drop all the packets that were queued waiting for that address resolution operation to complete.

Depending on the timeout value for the mapping function this situation might result in the attacked router running out of memory, with the consequence that incoming legitimate packets would have to be dropped, or that legitimate packets already stored in the router's memory buffers might need to be dropped. Both of these situations would lead either to a complete Denial of Service or to a degradation of the network service.

A number of countermeasures are warranted for this vulnerability:

Firstly, nodes should enforce a limit on the maximum number of packets that are queued for the same destination address while the corresponding address resolution operation is being performed. Additionally, nodes should enforce a limit on the aggregate number of packets that are queued waiting for address resolution operations to complete.

At the point the mapping function times out, all the packets destined to the address that timed out should be discarded. In addition, a 'negative cache entry' might be kept in the module performing the matching function, so that for some amount of time the mapping function would return an error when the IPv6 module requests

resolution of some IPv6 address for which the mapping has recently

failed (i.e., timed out).

A proactive mitigation for this vulnerability would be that when a packet is received that requires an address resolution operation before the packet can be forwarded, the packet is dropped and the router is then engaged in the address resolution operation.

This is a common implementation strategy for IPv4 routers. In IPv4, it is common that when a packet is received that requires an ARP request to be performed (before the packet can be forwarded), the packet is dropped and the router is then engaged in the ARP procedure.

While similar issues exist in IPv4 networks, this problem is exacerbated in IPv6, as IPv6 subnets are typically much larger than their IPv4 counterparts. Therefore, an attacker could send a large number of packets, each destined to different IPv6 addresses corresponding to non-existent 'neighbor nodes' of the attacked router. In the event that the router implementation drops packets only when the address resolution operation times out, the DoS condition might persist, whereas it would have probably disappeared if all the malicious packets had been destined to the same IPv6 address.

That is, if all the attack packets had been destined to the same IPv6 address, the timeout of the address resolution operation for that IPv6 address could have resulted in all the attack packets to be dropped.

An attacker could also potentially perform a Denial-of-Service attack against a router by exhausting the number of entries in the Neighbor cache and/or the Destination cache. In order to perform this attack, an attacker would send a large number of packets, each destined to different IPv6 addresses corresponding to non-existent 'neighbor nodes' of the attacked router. Each of these attack packets would trigger an address-resolution operation at the attacked router. If the target router does not enforce any limits on the maximum number of entries in the Neighbor cache, this attack could result in a buffer overflow at the attacked router. On the other hand, if the router does enforce limits on the maximum number of entries in the neighbor cache, the packets sent by the attacker could result in the attacked router hitting the aforementioned limit, probably preventing legitimate entries to be added to the Neighbor cache, resulting in a Denial of Service to the corresponding nodes.

This situation has been experienced in production networks probably as a result of reconnaissance activity by attackers. That is, this

situation could not only indicate a deliberate Denial-of-Service attack against a router, but could also be side-effect of network reconnaissance (i.e., 'scanning') activities.

A number of mitigations are warranted for this vulnerability:

- Nodes should enforce a limit on the number of entries in the Neighbor cache.
- Nodes should implement an algorithm to reclaim Neighbor Cache entries when the limit on the number of entries is reached.
- Nodes should enforce a limit on the number of entries in the Neighbor Cache that have a reachability state of 'INCOMPLETE'. This limit should be much stricter than the general limit on the number of entries in the Neighbor Cache.
- Nodes should enforce a limit on the number of entries in the Destination cache.
- Nodes should implement an algorithm to reclaim Destination Cache entries when the limit on the maximum number of entries is reached.

Section 5.3 of [RFC4861] states that for the purpose of eliminating unused entries (i.e., garbage-collection) in the Neighbor cache, any Least Recently Used (LRU)-based policy that only reclaims entries that have not been used in some time should be adequate. Such LRU-based policy should also be enforced to reclaim entries in the Neighbor cache or the Destination Cache when the limit on the maximum number of entries is hit for the Neighbor cache or the Destination cache, respectively.

5.1. Interface initialization

As explained in Section 7.2.1 of [RFC4861], when a multicast-capable interface is enabled, the node must join the all-nodes multicast group on that interface, and the solicited-node multicast address corresponding to each of the addresses assigned to an interface. As discussed in the same section, nodes join and leave the solicited-node groups as the assigned addresses change over time.

As the solicited-node multicast address for a number of assigned addresses might be the same, nodes should ensure that a solicited-node multicast group is not left until all the addresses corresponding to that solicited-node group have been removed.

An implementation that incorrectly leaves a solicited-node multicast

group while there are addresses corresponding to that multicast group still in use might be subject of a Denial-of-Service attack (from a malicious node attached to the same link as the victim).

In order to perform such an attack, an attacker would first send an unsolicited Router Advertisement, announcing a prefix such that the victim node configures an address whose solicited-node multicast group is the same as some legitimately-configured address. The advertised prefix would have a Lifetime value that would cause the address to be removed in the short term. If the Neighbor Discovery implementation of the victim system does not ensure that a solicited-node multicast group is left only when the last address corresponding to that solicited-node multicast group is removed, the victim might incorrectly leave the aforementioned solicited-node multicast group. As a result, the victim system would be unable to respond to Neighbor Solicitation messages for the target address, and therefore the aforementioned address would become unreachable.

5.2. Receipt of Neighbor Solicitation messages

As stated in Section 7.2.3, if a Neighbor Solicitation is received and an entry already exists in the Neighbor Cache for the IPv6 Source Address of the solicitation with a cached link-layer address that is different from the one in the received Source Link-Layer option, the cached address should be replaced by the received address (and the entry's reachability state must be set to STALE).

If an entry does not exist for the corresponding Target Address, a new entry is added to the Neighbor Cache, and its reachability state is set to STALE.

While this allows for improved responsiveness in the event the link-layer address corresponding to an IPv6 address changes, it also means that an attacker could easily override the mapping from IPv6 address to link-layer address.

An attacker attached to the same link as the victim system could craft a Neighbor Solicitation message with a forged IPv6 Source Address, and send it to the victim system, thus illegitimately

causing the victim to update its Neighbor Cache. The attacker could then send a Neighbor Advertisement with the Solicited flag set, thus causing the reachability state of the neighbor cache entry to be set to REACHABLE.

As a result, the attacker could cause traffic meant from the victim to the forged IPv6 address to be directed to any local system (i.e., attached to the same network link).

6. Vulnerability analysis

This section summarizes the security implications that arise from the Neighbor Discovery mechanisms in IPv6. The following vulnerabilities have been identified:

- Denial of Service: communication is prevented between legitimate nodes
- Performance-degrading: the performance of communication between legitimate nodes is reduced
- Traffic hijacking: traffic is illegitimately redirected to a node operated by an attacker

[RFC3756] provides a good security assessment of the Neighbor Discovery mechanisms. The following subs-sections summarize the results in [RFC3756], and also identify new vulnerabilities and specific attack vectors not present in that document.

Some of the vulnerabilities discussed in the following sub-sections involve an attacker sending Router Advertisement messages. [RFC6104] analyzes the problem of Rogue IPv6 Router Advertisements, and discuss a number of possible solutions. [RFC6105] discusses a specific solution to this problem based on layer-2 filtering, known as 'RA-Guard'. However, as discussed in [I-D.ietf-v6ops-ra-guard-implementation], some popular RA-Guard implementations can be easily circumvented by leveraging IPv6 extension headers.

[SI6-Toolkit] is a complete complete IPv6 toolkit that can be employed to exploit all the vulnerabilities discussed in the

following subsections. [THC-IPv6] includes 'proof of concept' tools of some of the vulnerabilities discussed in the following subsections. [vanHauser2006] is a presentation about this tool set.

[Beck2007b] analyzes the use of Neighbor Discovery for Operating System detection. However, the results seem to indicate that Neighbor Discovery is not an attractive means for Operating System detection when compared to other techniques such as those described in [CPNI-TCP]. Therefore, the 'information leakage' resulting from Neighbor Discovery, while possible, is not included in the threat analysis present in the following subsections.

6.1. Denial of Service

6.1.1. Neighbor Cache poisoning

Router Solicitation, Router Advertisement, Neighbor Solicitation and Neighbor Advertisement messages can be exploited to maliciously poison the Neighbor Cache of a victim node such that an IPv6 address maps into a non-existent link-layer address. As a result, traffic meant to the victim address would be 'black-holed' as a result of sending it to a non-existent link-layer address.

In the case of Router Solicitation, Router Advertisement, and Neighbor Solicitation messages, a source link-layer address would be employed. In the case of Neighbor Advertisement messages, a target link-layer address would be used instead.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link to which the target node is attached, or control a node attached to that network link (e.g., compromise such a node). However, it could be possible that as a result of tunnelling mechanisms, an attacker could perform these attacks remotely.

This attack could be mitigated by not overwriting the link-layer address of an existing Neighbor Cache entry when a source link-layer address option or a target link-layer address option is received. The mapping of IPv6 address to link-layer address would be updated

only if Neighbor Unreachability Detection (NUD) first removes the corresponding Neighbor Cache entry, and then a Neighbor Discovery message updates the mapping.

Furthermore, some monitoring tools exist that detect some possible exploitation of Neighbor Discovery and Neighbor Advertisement messages. NDPMon [NDPMon] is an example of such a monitoring tool (which is similar to the IPv4 arpwatch tool [arpwatch]). [Beck2007] is a paper about the aforementioned tool.

6.1.2. Tampering with Duplicate Address Detection (DAD)

The Duplicate Address Detection (DAD) mechanism is used to ensure that a node does configure for itself an address that is already in use.

An attacker could simply listen to Neighbor Solicitations sent as part of the DAD mechanism, and respond with crafted Neighbor Advertisements, thus causing the victim node to consider the address to be already in use, and thus preventing it from configuring the address for future use.

Additionally, an attacker could respond to Neighbor Solicitations

sent as part of the DAD mechanism with a Neighbor Solicitation for the same IPv6 address. The legitimate node performing DAD would consider this a collision and would cease to solicit that address (and possibly select and perform DAD for some alternative address).

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

Layer-2 switches could filter Neighbor Advertisement messages based on previous knowledge of the link-layer addresses recently in use at each port.

6.1.3. Tampering with Neighbor Unreachability Detection (NUD)

The Neighbor Unreachability Detection (NUD) mechanism is used to detect unreachable neighbors and cause the corresponding entries in

the Neighbor Cache to be eliminated. When an unreachable neighbor is detected and the corresponding Neighbor Cache entry is removed, a node has the chance to e.g., perform next-hop determination.

In order for a neighbor to be considered reachable, NUD uses reachability indications from upper-layer protocols. In the absence of reachability indications from an upper layer (e.g., from TCP's Acknowledgements), NUD employs solicited unicast Neighbor Solicitations to confirm the reachability of a Neighbor.

An attacker could snoop traffic and respond to the solicited Neighbor Solicitation messages being used for the purpose of NUD, thus preventing victim nodes from detecting that the impersonated node is unreachable. As a result, those 'victim' nodes would continue sending packets to the unreachable node, instead of e.g., performing first-hop determination to find an alternative working router.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

Nodes could require the link-layer source address of solicited Neighbor Advertisements being employed for NUD to be the same as that stored in the Neighbor Cache entry for which NUD is being performed. With this requirement in place, layer-2 switches could filter Neighbor Advertisement messages according to their source link-layer address, based on previous knowledge of the link-layer addresses recently in use at each port.

It should be noted that this recommendation should not be enforced in more complex networks in which VRRP [RFC5798] or custom redundancy protocols are employed.

This would mitigate the tampering with NUD that employs Neighbor Advertisement messages. However, it should be noted that an attacker might still tamper with NUD by forging upper-layer packets such as TCP Acknowledgements.

6.1.4. Rogue Router

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, advertising a non-existent system as a default router.

As a result, hosts honouring the aforementioned Router Advertisements would use the advertised rogue router as a default router, and as a result their packets would be black-holed.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node). As described in [RFC3756], this vulnerability could be mitigated by preferring existing routers over new ones.

Additionally, layer-2 switches could possibly allow Router Advertisements messages to be sent only from specific ports.

[RFC6104] analyzes the problem of Rogue IPv6 Router Advertisements, and discusses a number of possible solutions. [RFC6105] discusses a specific solution to this problem based on layer-2 filtering, known as 'RA-Guard'. However, as discussed in [I-D.ietf-v6ops-ra-guard-implementation], some popular RA-Guard implementations can be easily circumvented by leveraging IPv6 extension headers. [CVE-2011-2395] is a vulnerability advisory about this issue.

[SI6-Toolkit] is a complete complete IPv6 toolkit that can be employed to circumvent the aforementioned RA-Guard implementations.

6.1.5. Parameter spoofing

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, advertising a legitimate default router, but malicious network parameters.

For example, an attacker could advertise a very small Cur Hop Limit value, thus causing packets to be discarded before reaching their intended destination.

An attacker could also advertise an incorrect link MTU (either very small or very large) possibly preventing packets from being sent on the corresponding link and/or causing pathological behaviour at the victim nodes.

Finally, an attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, sending Router Advertisements with the M and/or the O bits set, thus possibly causing the victim nodes to engage in managed address configuration when such service is not present (e.g., there is no DHCP server) or when the attacker operates a malicious DHCP server.

In the former scenario, address configuration would fail, as a result of the non-existing DHCP server. In the latter scenario, an attacker could operate a malicious DHCP server to override IPv6's stateless configuration.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

As with other attacks based on Router Advertisement messages, they could be mitigated if Layer-2 switches allow Router Advertisements messages to be sent only from specific ports.

6.1.6. Bogus on-link prefixes

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, advertising bogus prefixes for on-link determination.

As a result, nodes belonging to the aforementioned prefixes would be considered on-link, and packets destined to them would not be relayed to a first-hop router. As a result, the victim nodes (i.e., those receiving the crafted Router Advertisements) would perform Neighbor Discovery for the intended destination, and when ND failed, the packets would be discarded.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network-link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

As mentioned in [RFC3756] host implementations could mitigate the impact of this attack by requiring the advertised prefixes to be at least /64s.

This would prevent an attacker from affecting a much larger portion of the IPv6 address space by e.g. sending a Router Advertisement that advertises the prefix ::/0 to be 'on-link'.

As with other attacks based on Router Advertisement messages, they could be mitigated if Layer-2 switches allow Router Advertisements messages to be sent only from specific ports.

6.1.7. Bogus address configuration prefixes

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, illegitimately advertising IPv6 prefixes for stateless address auto-configuration (SLAAC). This prefixes could either be bogon prefixes or prefixes owned by a remote site. An attacker could cause victim systems to configure IPv6 addresses using prefixes that would cause the resulting traffic to be black-holed.

This would cause the receiving nodes to configure their addresses with those bogus prefixes, and as a result, the resulting traffic would possibly be filtered by the network (e.g., if network egress-filtering is in place). Even if the outgoing packets were not filtered, these victim systems would not receive the return traffic, as the return traffic would either be filtered (in the case of bogon prefixes) or delivered to the legitimate destination (in the case of prefixes owned by some other party).

In order for an attacker to successfully perform this attack, he would need to be attached to the same network-link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

As with other attacks based on Router Advertisement messages, they could be mitigated if Layer-2 switches allow Router Advertisements messages to be sent only from specific ports.

6.1.8. Disabling routers

An attacker could send crafted Router Advertisements, Neighbor Advertisements, or Router Solicitations to cause the receiving nodes to remove the impersonated router from the router list.

In current implementations, if there are no default routers, a packet

destined to an off-link node will elicit an ICMPv6 Destination

Unreachable error message. In the case of legacy implementations compliant with [RFC2461], if there are no default routers, the Destination Address would be assumed to be 'on-link', and the victim would perform Neighbor Discovery for the destination address in the hope of delivering the packet on the local link.

In the case of the Router Advertisements vector, an attacker would send unsolicited Router Advertisements with a Preferred Lifetime equal to 0 or to some other small value, thus causing the receiving nodes to remove the impersonated router from the default router list.

Alternatively, an attacker could send forged Neighbor Advertisements (either solicited or unsolicited) with the Router flag set to 0, thus causing the impersonated router to be removed from the default router list.

Receiving nodes would assume the impersonated router has ceased to be a router and has changed to functioning only as a host.

As a third option, an attacker could send a forged Router Solicitation message to a router on the local network link, to cause the victim to remove the impersonated router from the router list. This attack vector is discussed in more detail in Section 4.1.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

Some IPv6 networks employ the 'RA-Guard' mechanism specified in [RFC6105] as the first line of defence against RA-based attack vectors. However, However, as discussed in [I-D.ietf-v6ops-ra-guard-implementation], some popular RA-Guard implementations can be easily circumvented by leveraging IPv6 extension headers. [CVE-2011-2395] is a vulnerability advisory about this issue.

[SI6-Toolkit] is a complete complete IPv6 toolkit that can be employed to circumvent the aforementioned RA-Guard implementations.

The rest of the attack vectors discussed in this section could possibly be mitigated with a more advanced Layer-2 filtering.

6.1.9. Tampering with 'on-link determination'

Section 2.1 of [RFC4861] states that a node considers an address to be on-link if:

- it is covered by one of the link's prefixes (e.g., as indicated by the on-link flag in the Prefix Information option), or
- a neighbouring router specifies the address as the target of a Redirect message, or
- a Neighbor Advertisement message is received for the (target) address, or
- any Neighbor Discovery message is received from the address.

As a result, some implementations create a Destination Cache entry for the Source Address of a Neighbor Discovery message (or for the Target Address of a Neighbor Advertisement message) when such a message is received, and mark the aforementioned address as 'on-link'.

This means in all traffic meant to the forged address will be delivered to the node identified in the corresponding Neighbor Cache entry (as the node will be considered to be on-link). If the corresponding Neighbor Cache entry maps the forged address into a non-existent or malicious node, all traffic can be black-holed, thus leading to a DoS scenario.

[RFC5942] updates [RFC4861], removing the third and fourth bullets in the above list. This means that receipt of ND messages must not result in the Source Address of the ND message or the Target Address of a Neighbor Advertisement message to be considered on-link (e.g., by modifying the Prefix List or by marking the corresponding Destination Cache entry as 'on-link').

[CVE-2008-2476] and [US-CERT2008] are vulnerability advisories about this issue.

Some IPv6 networks employ the 'RA-Guard' mechanism specified in [RFC6105] as the first line of defence against RA-based attack vectors. However, as discussed in [I-D.ietf-v6ops-ra-guard-implementation], some popular RA-Guard implementations can be easily circumvented by leveraging IPv6 extension headers. [CVE-2011-2395] is a vulnerability advisory about this issue.

[SI6-Toolkit] is a complete complete IPv6 toolkit that can be employed to circumvent the aforementioned RA-Guard implementations.

[I-D.ietf-6man-nd-extension-headers] updates [RFC3971] and [RFC4861], deprecating the use of fragmentation with Neighbor Discovery, such

that layer-2 filtering and Neighbor Discovery monitoring become feasible.

6.1.10. Introducing forwarding loops at routers

As discussed in Section 3.6.2 of this document, if broadcast or multicast addresses were allowed in source link-layer address options or in target link-layer address options, traffic directed to a victim IPv6 address would be sent to such broadcast or multicast IPv6 address.

Consider the following network scenario:

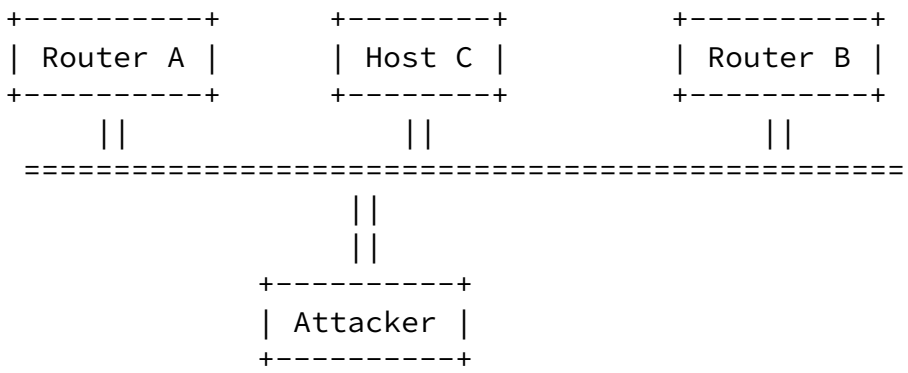


Figure 15: Example network scenario for forwarding loop

An attacker could poison the neighbor cache of Router A and the neighbor cache of Router B, such that the IPv6 address of Host C maps to the Ethernet broadcast address (ff:ff:ff:ff:ff:ff). Afterwards, he could send a packet to the Ethernet broadcast address (ff:ff:ff:ff:ff:ff), with an IPv6 Destination Address equal to the IPv6 address of Host C. Upon receiving the packet, both Router A and Router C would decrement the Hop Limit of the packet, and would resend it to the Ethernet broadcast address. As a result, both Router A and Router B would now receive two copies of the same packet (one sent by Router A, and another sent by Router B). This would result in a 'chain reaction' that would only disappear when the Hop Limit of each of the packets is decremented to 0. The total number of packets, for a general scenario in which multiple routers are present on the link and are subject of the aforementioned neighbor cache poisoning attack, and the attacker sends the initial attack packet with an arbitrary Hop Limit (possibly 255 to get the maximum amplification factor) is:

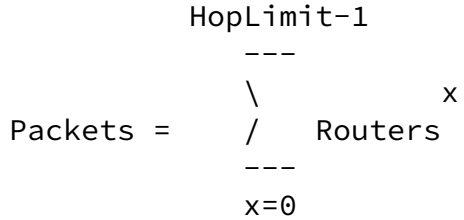


Figure 16: Maximum amplification factor

This equation does not take into account neither the possible ICMPv6 Redirect messages that each of the Routers could send, nor the possible ICMPv6 'time exceeded in transit' error messages that each of the routers could possibly send to the Source Address of the packet when each of the 'copies' of the original packet is discarded as a result of their Hop Limit being decremented to 0.

As discussed in Section 3.6.2 of this document, neither broadcast nor multicast addresses should be allowed in source link-layer address and target link-layer address options. An additional mitigation

would be for routers to not forward IPv6 packets on the same interface if the link-layer destination address of the packet was a broadcast or multicast address.

It is also possible to introduce a forwarding loop at a router by poisoning its neighbor cache such that a victim IPv6 address (considered to be on-link) maps to one of the attacked router's link-layer addresses. An attacker could poison the neighbor cache of the target router as described, and then send a packet to the attacked router with the IPv6 Destination Address set to the victim address. Upon receipt of the packet, the router would decrement the Hop Limit, and 'forward' the packet to its own link-layer address. This would result in a loop, with the target router processing the packet 'Hop Limit' times (where 'Hop Limit' is the value used for the Hop Limit field of the original packet).

6.1.11. Tampering with a Neighbor Discovery implementation

The Neighbor Discovery specification describes conceptual data structures such as the Neighbor Cache and the Destination Cache, which grow as a result of each entry that is created. Additionally, there are other structures such as the list of configured IPv6 addresses, the list of Recursive DNS Servers, etc., that also grow for each entry that is created in them.

As discussed throughout Section 5 of this document, an implementation should enforce limits on the maximum number of entries in these structures. Failure in enforcing such limits could result in buffer overflows or memory exhaustion.

FreeBSD 9.0 and NetBSD 5.1 fail to enforce limits on the number of entries in the IPv6 routing table, on the number of entries in the Neighbor Cache, on the number of entries in the Default Router List, and on the number of configured IPv6 addresses. Therefore they are vulnerable to multiple Denial of Service attacks.

Many versions of Windows that support IPv6 fail to enforce limits on the number of entries in the IPv6 routing table, on the maximum number of configured addresses, and on the number of entries in the Neighbor Cache. Therefore, these structures could be exploited for performing a Denial of Service attack. [Win-Update] describes an update has been made available for Windows 7 and

Windows Server 2008 R2 to limit the number of configured addresses and the number of routing table entries on a per-interface basis.

Linux 2.6.38-10 does enforce a limit on the number of entries in the Default router list. However, this limit itself could be leveraged for performing a Denial of Service attack, by causing the Default router list to become full of malicious/spurious entries before a legitimate entry can be added. As a result, the system would be unable to configure a legitimate default router, even if a legitimate Router Advertisement is received at some point later.

An attacker attached to the same network link as the target node can stress most of these data structures by sending a large number of the appropriate Neighbor Discovery options (e.g., RDNSS or Prefix Information options in Router Advertisement messages, etc.) as has been shown by e.g. [CVE-2010-4669].

Other structures (such as the Neighbor Cache or the Destination Cache) can be stressed by sending packets with forged addresses to the target node. For example, an attacker could send any packets that would elicit a response from the destination system with forged IPv6 Source Address that is assumed to be 'on-link' by the target system. In order for the target node to respond to those packets, it would have to create the necessary entries in the Destination Cache and in the Neighbor Cache. If the target implementation does not enforce limits on the maximum number of entries in each of those data structures, the attack may result in buffer overflows or kernel system memory exhaustion.

It is interesting to note that this attack vector could also be exploited by an attacker located in a remote site, unless ingress and/or egress filtering are in place.

[NISCC2006b] discusses ingress and egress filtering.

6.1.12. Tampering with a Neighbor Discovery router implementation from a remote site

A remote attacker could potentially perform a Denial-of-Service (DoS) attack against a router by sending packets to different IPv6 addresses considered on-link at one of the network links to which the target router is attached. Each of these packets would engage the target router in neighbor discovery for each of those addresses, probably preventing the router from performing neighbor discovery for legitimate packets aimed at existing nodes.

This problem would be exacerbated if an implementation queues in memory those packets that are destined to an IPv6 address for which address resolution is being performed. See Section 5 of this document for a thorough description of this issue.

One important difference between this attack vector and the ones described in the previous subsections is that in order for an attacker to successfully perform this attack, he does not need to be attached to the same network link to which the target router is attached.

A possible mitigation for this attack would be to enforce a limit on the maximum number of entries in the Neighbor Cache that are in the 'INCOMPLETE' state. This limit should be stricter than the overall limit on the maximum number of entries in the Neighbor Cache.

A Neighbor Cache entry is in the 'INCOMPLETE' state if a Neighbor Advertisement message has never been received for the corresponding IPv6 address since the entry was created.

It should be noted that this is an implementation issue rather than a protocol-based vulnerability. However, a number of implementations have been found to be vulnerable to this attack.

It is also worth noting that this attack does not require an attacker to forge the IPv6 Source Address of the 'malicious' packets. Therefore, mechanisms such as 'ingress filtering' do not provide any mitigation for this attack.

Section 6.1.11 describes another attack vector for stressing the Neighbor Cache (and the Destination cache) of both host and router implementations.

6.2. Performance degrading

6.2.1. Parameter spoofing

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, advertising a legitimate default router, but malicious network parameters.

An attacker could also advertise a small link MTU causing the victim nodes to enforce such a small MTU for the corresponding network link. This would increase the overhead (headers/data ratio), and possibly result in a packet-rate increase (if the same throughput is to be maintained). Additionally, this might also require the use of IPv6 fragmentation when data are to be transferred across this network link. This is a moderate version of the Denial-of-Service (DoS) attack discussed in Section 6.1.5 of this document.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

Some IPv6 networks employ the 'RA-Guard' mechanism specified in [RFC6105] as the first line of defence against RA-based attack vectors. However, as discussed in [I-D.ietf-v6ops-ra-guard-implementation], some popular RA-Guard implementations can be easily circumvented by leveraging IPv6 extension headers. [CVE-2011-2395] is a vulnerability advisory about this issue.

> [SI6-Toolkit] is a complete complete IPv6 toolkit that can be employed to circumvent the aforementioned RA-Guard implementations.

6.3. Traffic hijacking

6.3.1. Neighbor Cache poisoning

Neighbor Solicitation and Neighbor Advertisement messages can be exploited to maliciously poison the Neighbor Cache of a target node such that an IPv6 address maps into the link-layer address of a malicious node operated by an attacker. As a result, once the victim's Neighbor Cache is poisoned, the attacker would receive all traffic aimed at the victim node.

This is similar to the Denial-of-Service (DoS) attack described in Section 6.1.1 of this document, with the only difference being that in this case traffic would be directed to a node operated by the

attacker, rather than to a non-existent node.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

An attacker could also poison the Neighbor Cache of a target node mapping a victim IPv6 address to a multicast or broadcast link-layer address, such that he can receive a copy of those packets sent by the attacked node to the victim node. This specific attack vector is thoroughly discussed in Section 3.6.2 of this document.

The same mitigation techniques as described in Section 6.1.1 of this document apply to this attack-vector.

6.3.2. Rogue Router

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, advertising his own node as a default router.

This is similar to the Denial-of-Service (DoS) attack described in Section 6.1.4, with the only difference that in this case traffic would be directed to a node operated by the attacker, rather than to a non-existing node.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

The same mitigation techniques as described in Section 6.1.4 apply to this attack vector.

6.3.3. Bogus on-link prefixes

An attacker could either send unsolicited Router Advertisements and/or illegitimately respond to Router Solicitations, advertising bogus prefixes for on-link determination.

As a result, nodes belonging to the aforementioned prefixes would be

considered on-link, and packets destined to them would not be relayed to a first-hop router, but would instead be delivered on the local link. The victim nodes (i.e., those receiving the crafted Router Advertisements) would perform Neighbor Discovery for the intended destination, and the attacker could then respond with Neighbor Advertisements that advertise the link-layer address of his node, so

that packets are finally delivered to his malicious node.

In order for an attacker to successfully perform this attack, he would need to be attached to the same network link on which the attack is to be launched, or control a node attached to that network link (e.g., compromise such a node).

The same mitigation techniques as described in Section 6.1.6 apply to this attack vector.

6.3.4. Tampering with 'on-link determination'

This attack is similar to the Denial-of-Service (DoS) attack described in Section 6.1.10, with the only difference that for the purpose of traffic-hijacking, an attacker would make sure that the cached link-layer address of the Neighbor Cache entry corresponding to the victim address (the Source Address of the forged Neighbor Discovery message or the forged Target Address of the forged Neighbor Advertisement message) corresponds to the link-layer address of a node operated by the attacker.

As discussed in Section 6.1.9, [RFC5942] updates [RFC4861], such that this attack vector is eliminated. The same mitigations discussed in Section 6.1.9 of this document apply to mitigate this vulnerability.

[CVE-2008-2476] and [US-CERT2008] are vulnerability advisories about this issue.

6.4. Miscellaneous security issues

6.4.1. Detecting Sniffing Hosts

If a system reacts differently depending on whether the network interface is in promiscuous mode, this can be leveraged by an attacker that is on-link to infer whether the target node is in

promiscuous mode. Such a security issue has been found on many operating systems, where a packet with a multicast MAC address that is not being listened on by that target will be processed only if the receiving node is in promiscuous mode (i.e., "sniffing" the network). This test can be performed with any packet type, e.g. Neighbor Solicitation or Echo Request.

[CVE-2010-4562] is one vulnerability advisory about such an issue.

7. IANA Considerations

This document has no actions for IANA.

8. Security Considerations

This entire document is about security vulnerabilities that have been found popular Neighbor Discovery implementations, and other potential security issues that might be affecting existing implementations. This document not only discusses the aforementioned issues, but also provides implementation guidance such that these issues can be eliminated from the affected implementations and completely avoided or mitigated in any new Neighbor Discovery implementations.

The ultimate goal of this document is to help improve the overall maturity of Neighbor Discovery implementations, and to raise awareness about current security issues that might affect IPv6 networks.

9. Acknowledgements

Marc Heuse contributed text, edits, comments, and new vulnerabilities that were incorporated into this document.

The author would like to thank George Kargiotakis, who provided valuable comments on earlier versions of this document.

This document is based on the technical report "Security Assessment of the Internet Protocol version 6 (IPv6)" [CPNI-IPv6] authored by Fernando Gont on behalf of the UK Centre for the Protection of National Infrastructure (CPNI). The author would like to thank (in alphabetical order) Ran Atkinson, Fred Baker, Brian Carpenter, Roque Gagliano, Guillermo Gont, Alfred Hoenes, Qing Li, Neil Long, and

Pekka Savola, for providing valuable feedback on earlier versions of such document. Additionally, the author would like to thank (in alphabetical order) Ran Atkinson, Brian Carpenter, Joel M. Halpern, Robert Hinden, Pekka Savola, Fred Templin, and Ole Troan, who generously answered a number of questions when authoring the aforementioned document.

10. References

10.1. Normative References

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

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.

[RFC3122] Conta, A., "Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification", RFC 3122, June 2001.

[RFC3756] Nikander, P., Kempf, J., and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756, May 2004.

[RFC6275] Perkins, C., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, July 2011.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, November 2005.

[RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, April 2006.

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

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[RFC4943] Roy, S., Durand, A., and J. Paugh, "IPv6 Neighbor Discovery On-Link Assumption Considered Harmful", RFC 4943, September 2007.

[RFC6106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 6106, November 2010.

[RFC5798] Nadas, S., "Virtual Router Redundancy Protocol (VRRP)

Version 3 for IPv4 and IPv6", RFC 5798, March 2010.

[RFC5942] Singh, H., Beebee, W., and E. Nordmark, "IPv6 Subnet Model: The Relationship between Links and Subnet Prefixes", RFC 5942, July 2010.

[I-D.ietf-6man-nd-extension-headers] Gont, F., "Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery", draft-ietf-6man-nd-extension-headers-05 (work in progress), June 2013.

10.2. Informative References

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[RFC6104] Chown, T. and S. Venaas, "Rogue IPv6 Router Advertisement Problem Statement", RFC 6104, February 2011.

[RFC6105] Levy-Abegnoli, E., Van de Velde, G., Popoviciu, C., and J. Mohacsi, "IPv6 Router Advertisement Guard", RFC 6105, February 2011.

[I-D.ietf-v6ops-ra-guard-implementation] Gont, F., "Implementation Advice for IPv6 Router Advertisement Guard (RA-Guard)", draft-ietf-v6ops-ra-guard-implementation-07 (work in progress), November 2012.

[CPNI-IPv6] Gont, F., "Security Assessment of the Internet Protocol version 6 (IPv6)", UK Centre for the Protection of National Infrastructure, (available on request).

[CPNI-TCP] CPNI, "Security Assessment of the Transmission Control Protocol (TCP)", 2009, <http://www.gont.com.ar/papers/tn-03-09-security-assessment-TCP.pdf>.

[Hogg-Vyncke] Hogg, S. and E. Vyncke, "IPv6 Security", Cisco Press; 1 edition, 2008.

[Lecigne-Neville-Neil] Lecigne, C. and G. Neville-Neil, "Walking through FreeBSD IPv6 stack", 2006, <http://clem1.be/gimme/ipv6sec.pdf>.

[Beck2007]
Beck, F., Cholez, T., Festor, O., and I. Chrisment, "Monitoring the Neighbor Discovery Protocol", The Second International Workshop on IPv6 Today - Technology and Deployment - IPv6TD 2007, <http://hal.inria.fr/docs/00/15/35/58/PDF/IPv6TD07_beck.pdf>.

[Beck2007b]
Beck, F., Festor, O., and I. Chrisment, "IPv6 Neighbor Discovery Protocol based OS fingerprinting", INRIA Rapport Technique No 0345, 2007, <http://hal.archives-ouvertes.fr/docs/00/18/48/51/PDF/RT-0345.pdf>.

[NDPMon]
"NDPMon - IPv6 Neighbor Discovery Protocol Monitor", <http://ndpmon.sourceforge.net/>.

[arpwatch]
LBNL/NRG, "arpwatch tool", 2006, <http://ee.lbl.gov/>.

[NISCC2006b]
NISCC, "NISCC Technical Note 01/2006: Egress and Ingress Filtering", 2006, <http://www.niscc.gov.uk/niscc/docs/re-20060420-00294.pdf?lang=en>.

[vanHauser2006]
vanHauser, "Attacking the IPv6 Protocol Suite", EuSecWest 2006 Conference, <http://www.eusecwest.com/esw06/esw06-vanhauser.pdf>.

[SI6-Toolkit]
"SI6 Networks' IPv6 toolkit", <http://www.si6networks.com/tools/ipv6toolkit>.

[THC-IPv6]
"The Hacker's Choice IPv6 Attack Toolkit", <http://www.thc.org/thc-ipv6/>.

[CVE-2012-notyet]
CVE, "CVE-2012-notyet - entry is upcoming ... to be filled", 2012.

[CVE-2011-2391]

CVE, "CVE-2011-2391 - IPv6 Neighbor Discovery Protocol (NDP) implementations do not limit the rate of Neighbor Discovery messages processed", 2011.

[CVE-2008-2476]

CVE, "CVE-2008-2476 - IPv6 Neighbor Discovery Protocol (NDP) implementations do not validate the origin of Neighbor Discovery messages", 2008.

[CVE-2010-4669]
CVE, "CVE-2010-4669 - Neighbor Discovery (ND) protocol implementation in the IPv6 stack in Microsoft Windows allows attackers to cause a denial of service (CPU consumption and system hang) by sending many Router Advertisement (RA) messages with different source addresses", 2010.

[CVE-2011-2395]
CVE, "CVE-2011-2395 - Neighbor Discovery (ND) protocol implementation in Cisco IOS on unspecified switches allows attackers to bypass the Router Advertisement Guarding functionality via a fragmented IPv6 packets", 2011.

[CVE-2010-4562]
CVE, "CVE-2010-4562 - Microsoft Windows, when using IPv6, allows remote attackers to determine whether a host is sniffing the network by sending an ICMPv6 Echo Request to a multicast address and determining whether an Echo Reply is sent", 2010.

[US-CERT2008]
US-CERT, "US-CERT Vulnerability Note VU#472363: IPv6 implementations insecurely update Forwarding Information Base", 2008.

[Win-Update]
Microsoft, "An IPv6 readiness update is available for Windows 7 and for Windows Server 2008 R2", 2012.

Authors' Addresses

Fernando Gont
SI6 Networks / UTN-FRH
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

Phone: +54 11 4650 8472
Email: fgont@si6networks.com
URI: http://www.si6networks.com

Ronald P. Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, VA 20171
US

Phone: 571 250 5819
Email: rbonica@juniper.net

Will Liu
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com