Internet Engineering Task Force (IETF)
Request for Comments: 6971
Category: Experimental
ISSN: 2070-1721

U. Herberg, Ed.
Fujitsu
A. Cardenas
University of Texas at Dallas
T. Iwao
Fujitsu
M. Dow
Freescale
S. Cespedes
Icesi University
June 2013

Depth-First Forwarding (DFF) in Unreliable Networks

Abstract

This document specifies the Depth-First Forwarding (DFF) protocol for IPv6 networks, a data-forwarding mechanism that can increase reliability of data delivery in networks with dynamic topology and/or lossy links. The protocol operates entirely on the forwarding plane but may interact with the routing plane. DFF forwards data packets using a mechanism similar to a "depth-first search" for the destination of a packet. The routing plane may be informed of failures to deliver a packet or loops. This document specifies the DFF mechanism both for IPv6 networks (as specified in RFC 2460) and for "mesh-under" Low-Power Wireless Personal Area Networks (LoWPANs), as specified in RFC 4944. The design of DFF assumes that the underlying link layer provides means to detect if a packet has been successfully delivered to the Next Hop or not. It is applicable for networks with little traffic and is used for unicast transmissions only.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

This document defines an Experimental Protocol for the Internet community. This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6971.

Copyright Notice

Table of Contents

1. Introduction

This document specifies the Depth-First Forwarding (DFF) protocol for IPv6 networks, both for IPv6 forwarding [RFC2460] (henceforth denoted "route-over"), and also for "mesh-under" forwarding using the LoWPAN adaptation layer [RFC4944]. The protocol operates entirely on the forwarding plane but may interact with the routing plane. The purpose of DFF is to increase reliability of data delivery in networks with dynamic topologies and/or lossy links.

DFF forwards data packets using a "depth-first search" for the destination of the packets. DFF relies on an external neighborhood discovery mechanism that lists a router’s neighbors that may be attempted as Next Hops for a data packet. In addition, DFF may use information from the Routing Information Base (RIB) for deciding in which order to try to send the packet to the neighboring routers.

If the packet makes no forward progress using the first selected Next Hop, DFF will successively try all neighbors of the router. If none of the Next Hops successfully receives or forwards the packet, DFF returns the packet to the Previous Hop, which in turn tries to send it to alternate neighbors.

As network topologies do not necessarily form trees, loops can occur. Therefore, DFF contains a loop detection and avoidance mechanism.

DFF may provide information that may -- by a mechanism outside of this specification -- be used for updating the cost of routes in the RIB based on failed or successful delivery of packets through alternative Next Hops. Such information may also be used by a routing protocol.

DFF assumes that the underlying link layer provides means to detect if a packet has been successfully delivered to the Next Hop or not, is designed for networks with little traffic, and is used for unicast transmissions only.

1.1. Motivation

In networks with dynamic topologies and/or lossy links, even frequent exchanges of control messages between routers for updating the routing tables cannot guarantee that the routes correspond to the effective topology of the network at all times. Packets may not be delivered to their destination because the topology has changed since the last routing protocol update.

More frequent routing protocol updates can mitigate that problem to a certain extent; however, this requires additional signaling, consuming channel and router resources (e.g., when flooding control messages through the network). This is problematic in networks with lossy links, where further control traffic exchange can worsen the network stability because of collisions. Moreover, additional control traffic exchange may drain energy from battery-driven routers.

The data-forwarding mechanism specified in this document allows for forwarding data packets along alternate paths for increasing reliability of data delivery, using a depth-first search. The objective is to decrease the necessary control traffic overhead in the network and, at the same time, to increase delivery success rates.

As this specification is intended for experimentation, the mechanism is also specified for forwarding on the LoWPAN adaption layer (according to Section 11 of [RFC4944]), in addition to IPv6 forwarding as specified in [RFC2460]. Other than different header formats, the DFF mechanism for route-over and mesh-under is similar, and is therefore first defined in general and then more specifically for both IPv6 route-over forwarding (as specified in Section 13.1) and LoWPAN adaptation layer mesh-under (as specified in Section 13.2).

1.2. Experiments to Be Conducted

This document is presented as an Experimental specification that can increase reliability of data delivery in networks with dynamic topology and/or lossy links. It is anticipated that, once sufficient operational experience has been gained, this specification will be revised to progress it on to the Standards Track. This experiment is intended to be tried in networks that meet the applicability described in Section 3, and with the scope limitations set out in Section 14. While experimentation is encouraged in such networks, operators should exercise caution before attempting this experiment in other types of networks as the stability of interaction between DFF and routing in those networks has not been established.

Experience reports regarding DFF implementation and deployment are encouraged, particularly with respect to:

- Optimal values for the parameter P_HOLD_TIME, depending on the size of the network, the topology, and the amount of traffic originated per router. The longer a Processed Tuple is held, the more memory is consumed on a router. Moreover, if a tuple is held too long, a sequence number wrap-around may occur, and a new packet may have the same sequence number as one indicated in an old Processed Tuple. However, if the tuple is expired too soon (before the packet has completed its path to the destination), it may be mistakenly detected as a new packet instead of one already seen.

- Optimal values for the parameter MAX_HOP_LIMIT, depending on the size of the network, the topology, and how lossy the link layer is. MAX_HOP_LIMIT makes sure that packets do not unnecessarily traverse in the network; it may be used to limit the "detour" of packets that is acceptable. The value may also be issued on a per-packet basis if hop-count information is available from the RIB or routing protocol. In such a case, the Hop Limit for the packet may be a percentage (e.g., 200%) of the hop-count value indicated in the routing table.

- Optimal methods to increase the cost of a route when a loop or lost Layer 2 (L2) ACK is detected by DFF. While this is not specified as a normative part of this document, it may be of interest in an experiment to find good values of how much to increase link cost in the RIB or routing protocol.

- Performance of using DFF in combination with different routing protocols, such as reactive and proactive protocols. This also implies how routes are updated by the RIB or routing protocol when informed by DFF about loops or broken links.

2. Notation and Terminology

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

Additionally, this document uses the notation in Section 2.1 and the terminology in Section 2.2.

2.1. Notation

The following notations are used in this document:

List: A list of elements is defined as [] for an empty list, [element] for a list with one element, and [element1, element2, ...] for a list with multiple elements.

Concatenation of Lists: If List1 and List2 are lists, then List1@ List2 is a new list with all elements of List1 first, followed by all elements of List2.

Byte Order: All packet formats in this specification use network byte order (most significant octet first) for all fields. The most significant bit in an octet is numbered bit 0, and the least significant bit of an octet is numbered bit 7.

Assignment: a := b
An assignment operator, whereby the left side (a) is assigned the value of the right side (b).

Comparison: c = d
A comparison operator, returning true if the value of the left side (c) is equal to the value of the right side (d).

Flags: This specification uses multiple 1-bit flags. A value of ’0’ of a flag means ’false’; a value of ’1’ means ’true’.

2.2. Terminology

The terms "route-over" and "mesh-under", introduced in [RFC6775], are used in this document, where "route-over" is not only limited to IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) but also applies to general IPv6 networks.

Mesh-under: A topology where nodes are connected to a 6LoWPAN Border Router (6LBR) through a mesh using link-layer forwarding. Thus, in a mesh-under configuration, all IPv6 hosts in a LoWPAN are only one IP hop away from the 6LBR. This topology simulates the typical IP-subnet topology with one router with multiple nodes in the same subnet.

Route-over: A topology where hosts are connected to the 6LBR through the use of intermediate layer-3 (IP) routing. Here, hosts are typically multiple IP hops away from a 6LBR. The route-over topology typically consists of a 6LBR, a set of 6LoWPAN Routers (6LRs), and hosts.

The following terms are used in this document. As the DFF mechanism is specified both for route-over IPv6 and for the mesh-under LoWPAN adaptation layer, the terms are generally defined in this section, and then specifically mapped for each of the different modes of operation in Section 13.

Depth-First Search: "Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. One starts at the root (selecting some node as the root in the graph case) and explores as far as possible along each branch before backtracking" [DFS_wikipedia]. In this document, the algorithm for traversing a graph is applied to forwarding packets in a computer network, with nodes being routers.

Routing Information Base (RIB): A table stored in the user space of an operating system of a router or host. The table lists routes to network destinations, as well as associated metrics with these routes.

Mode of Operation (MoP): The DFF mechanism specified in this document can either be used as the "route-over" IPv6-forwarding mechanism (Mode of Operation: "route-over") or as the "mesh-under" LoWPAN adaptation layer (Mode of Operation: "mesh-under").

Packet: An IPv6 packet (for "route-over" MoP) or a "LoWPAN-encapsulated packet" (for "mesh-under" MoP), containing an IPv6 packet as payload.

Packet Header: An IPv6 extension header (for "route-over" MoP) or a LoWPAN header (for "mesh-under" MoP).

Address: An IPv6 address (for "route-over" MoP), or a 16-bit short or 64-bit Extended Unique Identifier (EUI-64) link-layer address (for "mesh-under" MoP).

Originator: The router that added the DFF header (specified in Section 7) to a packet.

Originator Address: An address of the Originator. According to [RFC6724], this address SHOULD be selected from the addresses that are configured on the interface that transmits the packet.

Destination: The router or host to which a packet is finally destined. In case this router or host is outside of the routing domain in which DFF is used, the destination is the router that removes the DFF header (specified in Section 7) from the packet. This case is described in Section 14.1.

Destination Address: An address to which the packet is sent.

Next Hop: An address of the Next Hop to which the packet is sent along the path to the destination.

Previous Hop: The address of the previous-hop router from which a packet has been received. In case the packet has been received by a router from outside of the routing domain where DFF is used (i.e., no DFF header is contained in the packet), the Originator Address of the router adding the DFF header to the packet is used as the Previous Hop.

Hop Limit: An upper bound denoting how many times the packet may be forwarded.

3. Applicability Statement

This document specifies DFF, a packet-forwarding mechanism intended for use in networks with dynamic topology and/or lossy links with the purpose of increasing reliability of data delivery. The protocol’s applicability is determined by its characteristics, which are that this protocol:

- Is applicable for use in IPv6 networks, either as a "route-over" forwarding mechanism using IPv6 [RFC2460], or as a "mesh-under" forwarding mechanism using the frame format for transmission of IPv6 packets, as defined in [RFC4944].

- Assumes addresses used in the network are either IPv6 addresses (if the protocol is used as "route-over"), or 16-bit short or EUI-64 link-layer addresses, as specified in [RFC4944], if the protocol is used as "mesh-under". In "mesh-under" mode, mixed 16-bit and EUI-64 addresses within one DFF routing domain are allowed (if they conform with [RFC4944]), as long as DFF is limited to use within one PAN (Personal Area Network). It is assumed that the "route-over" mode and "mesh-under" mode are mutually exclusive in the same routing domain.

- Assumes that the underlying link layer provides means to detect if a packet has been successfully delivered to the Next Hop or not (e.g., by L2 ACK messages). Examples for such underlying link layers are specified in IEEE 802.15.4 and IEEE 802.11.

- Is applicable in networks with lossy links and/or with a dynamic topology. In networks with very stable links and fixed topology, DFF will not bring any benefit (but also will not be harmful, other than the additional overhead for the packet header).

- Works in a completely distributed manner and does not depend on any central entity.

- Is applicable for networks with little traffic in terms of numbers of packets per second, since each recently forwarded packet increases the state on a router. The amount of traffic per time that is supported by DFF depends on the memory resources of the router running DFF, the density of the network, the loss rate of the channel, and the maximum Hop Limit for each packet: for each recently seen packet, a list of Next Hops that the packet has been sent to is stored in memory. The stored entries can be deleted after an expiration time, so that only recently received packets require storage on the router. Implementations are advised to measure and report rates of packets in the network, and also to report memory usage. Thus, operators can determine memory exhaustion because of growing information sets or problems because of too rapid sequence-number wrap-around.

- Is applicable for dense topologies with multiple paths between each source and each destination. Certain topologies are less suitable for DFF: topologies that can be partitioned by the removal of a single router or link, topologies with multiple stub routers that each have a single link to the network, topologies with only a single path to a destination, or topologies where the "detour" that a packet makes during the depth-first search in order to reach the destination would be too long. Note that the number of retransmissions of a packet that stipulate a "too long" path depends on the underlying link layer (capacity and probability of packet loss), as well as how much bandwidth is required for data traffic by applications running in the network. In such topologies, the packet may never reach the destination; therefore, unnecessary transmissions of data packets may occur until the Hop Limit of the packet reaches zero, and the packet is dropped. This may consume channel and router resources.

- Is used for unicast transmissions only (not for anycast or multicast).

- Is for use within stub networks and for traffic between a router inside the routing domain in which DFF is used and a known border router. Examples of such networks are LoWPANs. Scope limitations are described in Section 14.

4. Protocol Overview and Functioning

When a packet is to be forwarded by a router using DFF, the router creates a list of candidate Next Hops for that packet. This list (created per packet) is ordered, and Section 11 provides recommendations on how to order the list, e.g., first listing Next Hops listed in the RIB, if available, ordered in increasing cost, followed by other neighbors provided by an external neighborhood discovery. DFF proceeds to forward the packet to the first Next Hop in the list. If the transmission was not successful (as determined by the underlying link layer) or if the packet was "returned" by a Next Hop to which it had been sent before, the router will try to forward the packet to the subsequent Next Hop on the list. A router "returns" a packet to the router from which it was originally received once it has unsuccessfully tried to forward the packet to all elements in the candidate Next Hop list. If the packet is eventually returned to the Originator of the packet, and after the

Originator has exhausted all of its Next Hops for the packet, the packet is dropped.

For each recently forwarded packet, a router running DFF stores information about the packet as an entry in an information set, denoted "Processed Set". Each entry in the Processed Set contains a sequence number, included in the packet header, identifying the packet. (Refer to Section 12 for further details on the sequence number.) Furthermore, the entry contains a list of Next Hops to which the packet has been sent. This list of recently forwarded packets also allows for avoiding loops when forwarding a packet. Entries in the Processed Set expire after a given expiration timeout and are removed.

4.1. Overview of Information Sets

This specification requires a single set on each router, the Processed Set. The Processed Set stores the sequence number, the Originator Address, the Previous Hop, and a list of Next Hops to which the packet has been sent, for each recently seen packet. Entries in the set are removed after a predefined timeout. Each time a packet is forwarded to a Next Hop, that Next Hop is added to the list of Next Hops of the entry for the packet.

Note that an implementation of this protocol may maintain the information of the Processed Set in the indicated form, or in any other organization that offers access to this information. In particular, it is not necessary to remove tuples from a set at the exact time indicated, only to behave as if the tuples were removed at that time.

In addition to the Processed Set, a list of symmetric neighbors must be provided by an external neighborhood discovery mechanism, or may be determined from the RIB (e.g., if the RIB provides routes to adjacent routers, and if these one-hop routes are verified to be symmetric).

4.2. Signaling Overview

Information is needed on a per-packet basis by a router that is running DFF and receives a packet. This information is encoded in the packet header that is specified in this document as the IPv6 Hop-by-Hop Options header and LoWPAN header, respectively, for the intended "route-over" and "mesh-under" Modes of Operation. This DFF header contains a sequence number used for uniquely identifying a packet and two flags, RET (for "return") and DUP (for "duplicate").

While a router successively tries sending a data packet to one or more of its neighbors, RET = 0. If none of the transmissions of the packet to the neighbors of a router have succeeded, the packet is returned to the router from which the packet was first received, indicated by setting the return flag (RET := 1). The RET flag is required to discern between a deliberately returned packet and a looping packet: if a router receives a packet with RET = 1 (and DUP = 0 or DUP = 1) that it has already forwarded, the packet was deliberately returned, and the router will continue to successively send the packet to routers from the candidate Next Hop list. If that packet has RET = 0, the router assumes that the packet is looping and returns it to the router from which it was last received. An external mechanism may use this information for increasing the route cost of the route to the destination using the Next Hop that resulted in the loop in the RIB or the routing protocol. It is out of scope of this document to specify such a mechanism. Note that once DUP is set to 1, loop detection is not possible any more as the flag is not reset any more. Therefore, a packet may loop if the RIBs of routers in the domain are inconsistent, until the Hop Limit has reached 0.

Whenever a packet transmission to a neighbor has failed (as determined by the underlying link layer, e.g., using L2 ACKs), the DUP flag is set in the packet header for the following transmissions. The rationale is that the packet may have been successfully received by the neighbor and only the L2 ACK has been lost, resulting in possible duplicates of the packet in the network. The DUP flag tags such a possible duplicate. The DUP flag is required to discern between a duplicated packet and a looping packet: if a router receives a packet with DUP = 1 (and RET = 0) that it has already forwarded, the packet is not considered looping and is successively forwarded to the next router from the candidate Next Hop list. If the received packet has DUP = 0 (and RET = 0), the router assumes that the packet is looping, sets RET := 1, and returns it to the Previous Hop. Again, an external mechanism may use this information for increasing route costs and/or informing the routing protocol.

The reason for not dropping received duplicated packets (with DUP = 1) is that a duplicated packet may be duplicated again during its path if another L2 ACK is lost. However, when DUP is already set to 1, it is not possible to discern the duplicate from the duplicate of the duplicate. As a consequence, loop detection is not possible after the second lost L2 ACK on the path of a packet. However, if duplicates are simply dropped, it is possible that the packet was actually a looping packet (and not a duplicate), and so the depth-first search would be interrupted.

5. Protocol Dependencies

DFF MAY use information from the Routing Information Base (RIB), specifically for determining an order of preference for which Next Hops a packet should be forwarded to (e.g., the packet may be forwarded first to neighbors that are listed in the RIB as Next Hops to the destination, preferring those with the lowest route cost). Section 11 provides recommendations about the order of preference for the Next Hops of a packet.

DFF MUST have access to a list of symmetric neighbors for each router; this list is provided by a neighborhood discovery protocol, such as the one defined in [RFC6130]. A neighborhood discovery protocol is not specified in this document.

6. Information Sets

This section specifies the information sets used by DFF.

6.1. Symmetric Neighbor List

DFF MUST have access to a list of addresses of symmetric neighbors of the router. This list can be provided by an external neighborhood discovery mechanism or, alternatively, may be determined from the RIB (e.g., if the RIB provides routes to adjacent routers, and if these one-hop routes are verified to be symmetric). The list of addresses of symmetric neighbors is not specified within this document. The addresses in the list are used to construct a list of candidate Next Hops for a packet, as specified in Section 11.

6.2. Processed Set

Each router maintains a Processed Set in order to support the loop detection functionality. The Processed Set lists sequence numbers of previously received packets, as well as a list of Next Hops to which the packet has been sent successively as part of the depth-first forwarding mechanism. To protect against this situation, it is recommended that an implementation retains the Processed Set in non-volatile storage if such is provided by the router.

The set consists of Processed Tuples

```
(P_orig_address, P_seq_number, P_prev_hop,
P_next_hop_neighbor_list, P_time)
```

where

P_orig_address is the Originator Address of the received packet;

P_seq_number is the sequence number of the received packet;

P_prev_hop is the address of the Previous Hop of the packet;

P_next_hop_neighbor_list is a list of addresses of Next Hops to which the packet has been sent previously, as part of the depth-first forwarding mechanism, as specified in Section 9.2;

P_time specifies when this tuple expires and MUST be removed.

The consequences when no, or not enough, non-volatile storage is available on a router (e.g., because of limited resources) or when an implementation chooses not to make the Processed Set persistent are that packets that are already in a loop caused by the routing protocol may continue to loop until the Hop Limit is exhausted. Non-looping packets may be sent to Next Hops that have already received the packet previously and will return the packet, leading to some unnecessary retransmissions. This effect is only temporary and applies only for packets already traversing the network.

7. Packet Header Fields

This section specifies the information required by DFF in the packet header. Note that, depending on whether DFF is used in the "route-over" MoP or in the "mesh-under" MoP, the DFF header is either an IPv6 Hop-by-Hop Options header (as specified in Section 13.1.2) or a LoWPAN header (as specified in Section 13.2.2). Sections 13.1.2 and 13.2.2 specify the precise order, format, and encoding of the fields that are listed in this section.

Version (VER) - This 2-bit value indicates the version of DFF that is used. This specification defines value ’00’. Packets with other values of the version MUST be forwarded using the route-over MoP and mesh-under MoP as defined in [RFC2460] and [RFC4944], respectively.

Duplicate (DUP) Packet Flag - This 1-bit flag is set in the DFF header of a packet when that packet is being retransmitted due to a signal from the link layer that the original transmission failed, as specified in Section 9.2. Once the flag is set to 1, it MUST NOT be modified by routers forwarding the packet.

Return (RET) Packet Flag - This 1-bit flag MUST be set to 1 prior to sending the packet back to the Previous Hop. Upon receiving a packet with RET = 1, and before sending it to a new candidate Next Hop, that flag MUST be set to 0, as specified in Section 9.2.

Sequence Number - A 16-bit field, containing an unsigned integer sequence number generated by the Originator, unique to each router for each packet to which the DFF has been added, as specified in Section 12. The Originator Address concatenated with the sequence number represents an identifier of previously seen data packets. Refer to Section 12 for further information about sequence numbers.

8. Protocol Parameters

The parameters used in this specification are listed in this section. These parameters are configurable, do not need to be stored in non-volatile storage, and can be varied by implementations at run-time. Default values for the parameters depend on the network size, topology, link layer, and traffic patterns. Part of the experimentation described in Section 1.2 is to determine suitable default values.

P_HOLD_TIME - Is the time period after which a newly created or modified Processed Tuple expires and MUST be deleted. An implementation SHOULD use a value for P_HOLD_TIME that is high enough that the Processed Tuple for a packet is still in memory on all forwarding routers while the packet is transiting the routing domain. The value SHOULD at least be MAX_HOP_LIMIT times the expected time to send a packet to a router on the same link. The value MUST be lower than the time it takes until the same sequence number is reached again after a wrap-around on the router identified by P_orig_address of the Processed Tuple.

MAX_HOP_LIMIT - Is the initial value of Hop Limit, and therefore the maximum number of times that a packet is forwarded in the routing domain. When choosing the value of MAX_HOP_LIMIT, the size of the network, the distance between source and destination in number of hops, and the maximum possible "detour" of a packet SHOULD be considered (compared to the shortest path). Such information MAY be used from the RIB, if provided.

9. Data Packet Generation and Processing

The following sections specify the process of handling a packet entering the DFF routing domain, i.e., without a DFF header (Section 9.1), as well as forwarding a data packet from another router running DFF (Section 9.2).

9.1. Data Packets Entering the DFF Routing Domain

This section applies for any data packets upon their first entry into a routing domain in which DFF is used. This occurs when a new data packet is generated on this router, or when a data packet is forwarded from outside the routing domain (i.e., from a host attached to this router or from a router outside the routing domain in which DFF is used). Before such a data packet (henceforth denoted "current packet") is transmitted, the following steps MUST be executed:

1. If required, encapsulate the packet, as specified in Section 14.

2. Add the DFF header to the current packet (to the outer header if the packet has been encapsulated) with:

   * DUP := 0;

   * RET := 0;

   * Sequence Number := a new sequence number of the packet (as specified in Section 12).

3. Check that the packet does not exceed the MTU, as specified in Section 15. In case it does, execute the procedures listed in Section 15 and do not further process the packet.

4. Select the Next Hop (henceforth denoted "next_hop") for the current packet, as specified in Section 11.

5. Add a Processed Tuple to the Processed Set with:

   * P_orig_address := the Originator Address of the current packet;

   * P_seq_number := the sequence number of the current packet;

   * P_prev_hop := the Originator Address of the current packet;

   * P_next_hop_neighbor_list := [next_hop];

   * P_time := current time + P_HOLD_TIME.

6. Pass the current packet to the underlying link layer for transmission to next_hop. If the transmission fails (as determined by the link layer), the procedures in Section 10 MUST be executed.

9.2. Data Packet Processing

When a packet (henceforth denoted the "current packet") is received by a router, the following tasks MUST be performed:

1. If the packet header is malformed (i.e., the header format is not as expected by this specification), drop the packet.

2. Otherwise, if the Destination Address of the packet matches an address of an interface of this router, deliver the packet to upper layers and do not further process the packet, as specified below.

3. Decrement the value of the Hop Limit field by one (1).

4. Drop the packet if Hop Limit is decremented to zero and do not further process the packet, as specified below.

5. If no Processed Tuple (henceforth denoted the "current tuple") exists in the Processed Set, where both of the following conditions are true:

   + P_orig_address = the Originator Address of the current packet, AND;

   + P_seq_number = the sequence number of the current packet.

   Then:

   1. Add a Processed Tuple (henceforth denoted the "current tuple") with:

      + P_orig_address := the Originator Address of the current packet;

      + P_seq_number := the sequence number of the current packet;

      + P_prev_hop := the Previous Hop Address of the current packet;

      + P_next_hop_neighbor_list := [];

      + P_time := current time + P_HOLD_TIME.

   2. Set RET to 0 in the DFF header.

   3. Select the Next Hop (henceforth denoted "next_hop") for the current packet, as specified in Section 11.

```
       4.  P_next_hop_neighbor_list := P_next_hop_neighbor_list@
           [next_hop].

       5.  Pass the current packet to the underlying link layer for
           transmission to next_hop.  If the transmission fails (as
           determined by the link layer), the procedures in Section 10
           MUST be executed.

   6.  Otherwise, if a tuple exists:

       1.  If the return flag of the current packet is not set (RET = 0)
           (i.e., a loop has been detected):

           1.  Set RET := 1.

           2.  Pass the current packet to the underlying link layer for
               transmission to the Previous Hop.

       2.  Otherwise, if the return flag of the current packet is set
           (RET = 1):

           1.  If the Previous Hop of the packet is not contained in
               P_next_hop_neighbor_list of the current tuple, drop the
               packet.

           2.  If the Previous Hop of the packet (i.e., the address of
               the router from which the current packet has just been
               received) is equal to P_prev_hop of the current tuple
               (i.e., the address of the router from which the current
               packet has been first received), drop the packet.

           3.  Set RET := 0.

           4.  Select the Next Hop (henceforth denoted "next_hop") for
               the current packet, as specified in Section 11.

           5.  Modify the current tuple:

               -  P_next_hop_neighbor_list := P_next_hop_neighbor_list@
                  [next_hop];

               -  P_time := current time + P_HOLD_TIME.
```

6. If the selected Next Hop is equal to P_prev_hop of the current tuple, as specified in Section 11 (i.e., all candidate Next Hops have been unsuccessfully tried), set RET := 1. If this router (i.e., the router receiving the current packet) has the same address as the Originator Address of the current packet, drop the packet.

7. Pass the current packet to the underlying link layer for transmission to next_hop. If transmission fails (as determined by the link layer), the procedures in Section 10 MUST be executed.

10. Unsuccessful Packet Transmission

DFF requires that the underlying link layer provides information as to whether a packet is successfully received by the Next Hop. Absence of such a signal is interpreted as a delivery failure of the packet (henceforth denoted the "current packet"). Note that the underlying link layer MAY retry sending the packet multiple times (e.g., using exponential back-off) before determining that the packet has not been successfully received by the Next Hop. The following steps are executed when a delivery failure occurs and Section 9 requests that they be executed.

1. Set the DUP flag of the DFF header of the current packet to 1.

2. Select the Next Hop (henceforth denoted "next_hop") for the current packet, as specified in Section 11.

3. Find the Processed Tuple (the "current tuple") in the Processed Set with:

   + P_orig_address = the Originator Address of the current packet, AND;

   + P_seq_number = the sequence number of the current packet.

4. If no current tuple is found, drop the packet.

5. Otherwise, modify the current tuple:

   * P_next_hop_neighbor_list := P_next_hop_neighbor_list@ [next_hop];

   * P_time := current time + P_HOLD_TIME.

6. If the selected next_hop is equal to P_prev_hop of the current tuple, as specified in Section 11 (i.e., all neighbors have been unsuccessfully tried), then:

   * RET := 1

   * Decrement the value of the Hop Limit field by one (1). Drop the packet if the Hop Limit is decremented to zero.

7. Otherwise

   * RET := 0

8. Transmit the current packet to next_hop. If transmission fails (as determined by the link layer), and if the next_hop does not equal P_prev_hop from the current tuple, the procedures in Section 10 MUST be executed.

11. Determining the Next Hop for a Packet

When forwarding a packet, a router determines a valid Next Hop for that packet, as specified in this section. As a Processed Tuple either existed when receiving the packet (henceforth denoted the "current packet") or was created, it can be assumed that the Processed Tuple for that packet (henceforth denoted the "current tuple") is available.

The Next Hop is chosen from a list of candidate Next Hops in order of decreasing priority. This list is created per packet. The maximum candidate Next Hop list for a packet contains all the neighbors of the router (as determined from an external neighborhood discovery process), except for the Previous Hop of the current packet. A smaller list MAY be used, if desired, and the exact selection of the size of the candidate Next Hop list is a local decision that is made in each router and does not affect interoperability. Selecting a smaller list may reduce the path length of a packet traversing the network and reduce the required state in the Processed Set, but it may result in valid paths that are not explored. If information from the RIB is used, then the candidate Next Hop list MUST contain at least the Next Hop indicated in the RIB as the Next Hop on the shortest path to the destination, and it SHOULD contain all Next Hops indicated to the RIB as Next Hops on paths to the destination. If a Next Hop from the RIB equals the Previous Hop of the current packet, it MUST NOT be added to the candidate Next Hop list.

The list MUST NOT contain addresses that are listed in P_next_hop_neighbor_list of the current tuple, in order to avoid sending the packet to the same neighbor multiple times. Moreover, an address MUST NOT appear more than once in the list, for the same reason. Also, addresses of an interface of this router MUST NOT be added to the list.

The list has an order of preference, where packets are first sent to the Next Hops at the top of the list during depth-first processing as specified in Sections 9.1 and 9.2. The following order is RECOMMENDED, with the elements listed on top having the highest preference:

1. The neighbor that is indicated in the RIB as the Next Hop on the shortest path to the destination of the current packet;

2. Other neighbors indicated in the RIB as Next Hops on the path to the destination of the current packet;

3. All other symmetric neighbors (except the Previous Hop of the current packet).

Additional information from the RIB or the list of symmetric neighbors (such as route cost or link quality) MAY be used for determining the order.

If the candidate Next Hop list created as specified in this section is empty, the selected Next Hop MUST be P_prev_hop of the current tuple; this case applies when returning the packet to the Previous Hop.

12. Sequence Numbers

Whenever a router generates a packet or forwards a packet on behalf of a host or a router outside the routing domain where DFF is used, a sequence number MUST be created and included in the DFF header. This sequence number MUST be unique locally on each router where it is created. A sequence number MUST start at 0 for the first packet to which the DFF header is added, and then increment by 1 for each new packet. The sequence number MUST NOT be greater than 65535 and MUST wrap around to 0.

13. Modes of Operation

DFF can be used either as the "route-over" IPv6-forwarding protocol, or alternatively as the "mesh-under" data-forwarding protocol for the LoWPAN adaptation layer [RFC4944]. Previous sections have specified the DFF mechanism in general; specific differences for each MoP are specified in this section.

13.1. Route-Over

This section maps the general terminology from Section 2.2 to the specific terminology when using the "route-over" MoP.

13.1.1. Mapping of DFF Terminology to IPv6 Terminology

The following terms are those listed in Section 2.2, and their meaning is explicitly defined when DFF is used in the "route-over" MoP:

Packet - An IPv6 packet, as specified in [RFC2460].

Packet Header - An IPv6 extension header, as specified in [RFC2460].

Address - An IPv6 address, as specified in [RFC4291].

Originator Address - The Originator Address corresponds to the Source Address field of the IPv6 header, as specified in [RFC2460].

Destination Address - The Destination Address corresponds to the destination field of the IPv6 header, as specified in [RFC2460].

Next Hop - The Next Hop is the IPv6 address of the node to which the packet is sent; the link-layer address from that IP address is resolved by a mechanism such as Neighbor Discovery (ND) [RFC4861]. The link-layer address is then used by L2 as the destination.

Previous Hop - The Previous Hop is the IPv6 address from the interface of the node from which the packet has been received.

Hop Limit - The Hop Limit corresponds to the Hop Limit field in the IPv6 header, as specified in [RFC2460].

13.1.2. Packet Format

In the "route-over" MoP, all IPv6 packets MUST conform with the format specified in [RFC2460].

The DFF header, as specified below, is an IPv6 Hop-by-Hop Options header, and is depicted in Figure 1 (where DUP is abbreviated to D, and RET is abbreviated to R because of the limited space in the figure). This document specifies a new option to be used inside the Hop-by-Hop Options header, which contains the DFF fields (DUP and RET flags and sequence number, as specified in Section 7).

[RFC6564] specifies:

> New options for the existing Hop-by-Hop Header SHOULD NOT be created or specified unless no alternative solution is feasible. Any proposal to create a new option for the existing Hop-by-Hop Header MUST include a detailed explanation of why the hop-by-hop behavior is absolutely essential in the document proposing the new option with hop-by-hop behavior.

[RFC6564] recommends to use destination headers instead of Hop-by-Hop Options headers. Destination headers are only read by the destination of an IPv6 packet, not by intermediate routers. However, the mechanism specified in this document relies on intermediate routers reading and editing the header. Specifically, the sequence number and the DUP and RET flags are read by each router running the DFF protocol. Modifying the DUP and RET flags is essential for this protocol to tag duplicate or returned packets. Without the DUP flag, a duplicate packet cannot be discerned from a looping packet, and without the RET flag, a returned packet cannot be discerned from a looping packet.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  |  OptTypeDFF   | OptDataLenDFF |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|VER|D|R|0|0|0|0|        Sequence Number        |      Pad1     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: IPv6 DFF Header

Field definitions of the DFF header are as follows:

Next Header - 8-bit selector. Identifies the type of header immediately following the Hop-by-Hop Options header, as specified in [RFC2460].

Hdr Ext Len - 8-bit unsigned integer. Length of the Hop-by-Hop Options header in 8-octet units, not including the first 8 octets, as specified in [RFC2460]. This value is set to 0 (zero).

OptTypeDFF - 8-bit identifier of the type of option, as specified in [RFC2460]. This value is set to IP_DFF. The two high-order bits of the option type MUST be set to ’11’, and the third bit is equal to ’1’. With these bits, according to [RFC2460], routers that do not understand this option on a received packet discard the packet and, only if the packet’s Destination Address was not a multicast address, send an ICMP Parameter Problem (Code 2) message to the packet’s Source Address, pointing to the unrecognized option type. Also, according to [RFC2460], the values within the option are expected to change en route.

OptDataLenDFF - 8-bit unsigned integer. Length of the option data field of this option, in octets, as specified in [RFC2460]. This value is set to 2 (two).

DFF fields - A 2-bit version field (abbreviated as VER); the DUP (abbreviated as D) and RET (abbreviated as R) flags follow after Mesh Forw, as specified in Section 13.2.2. The version specified in this document is ’00’. All other bits (besides VER, DUP, and RET) of this octet are reserved and MUST be set to 0.

Sequence Number - A 16-bit field, containing an unsigned integer sequence number, as specified in Section 7.

Pad1 - Since the Hop-by-Hop Options header must have a length that is a multiple of 8 octets, a Pad1 option is used, as specified in [RFC2460]. All bits of this octet are 0.

13.2. Mesh-Under

This section maps the general terminology from Section 2.2 to the specific terminology when using the "mesh-under" MoP.

13.2.1. Mapping of DFF Terminology to LoWPAN Terminology

The following terms are those listed in Section 2.2 (besides "Mode of Operation"), and their meaning is explicitly defined when DFF is used in the "mesh-under" MoP.

Packet - A "LoWPAN-encapsulated packet" (as specified in [RFC4944]), which contains an IPv6 packet as payload.

Packet Header - A LoWPAN header, as specified in [RFC4944].

Address - A 16-bit short or EUI-64 link-layer address, as specified in [RFC4944].

Originator Address - The Originator Address corresponds to the Originator Address field of the Mesh Addressing header, as specified in [RFC4944].

Destination Address - The Destination Address corresponds to the Final Destination field of the Mesh Addressing header, as specified in [RFC4944].

Next Hop - The Next Hop is the Destination Address of a frame containing a LoWPAN-encapsulated packet, as specified in [RFC4944].

Previous Hop - The Previous Hop is the Source Address of the frame containing a LoWPAN-encapsulated packet, as specified in [RFC4944].

Hop Limit - The Hop Limit corresponds to the Deep Hops Left field in the Mesh Addressing header, as specified in [RFC4944].

13.2.2. Packet Format

In the "mesh-under" MoP, all IPv6 packets MUST conform with the format specified in [RFC4944]. All data packets exchanged by routers using this specification MUST contain the Mesh Addressing header as part of the LoWPAN encapsulation, as specified in [RFC4944].

The DFF header, as specified below, MUST follow the Mesh Addressing header. After these two headers, any other LoWPAN header, e.g., header compression or fragmentation headers, MAY also be added before the actual payload. Figure 2 depicts the Mesh Addressing header defined in [RFC4944], and Figure 3 depicts the DFF header.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1 0|V|F|HopsLft| DeepHopsLeft  |orig. address, final address...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Mesh Addressing Header

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1| Mesh Forw |VER|D|R|0|0|0|0|       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Header for DFF Data Packets

Field definitions of the Mesh Addressing header are as specified in [RFC4944]. When adding that header to the LoWPAN encapsulation on the Originator, the fields of the Mesh Addressing header MUST be set to the following values:

- V := 0 if the Originator Address is an IEEE extended 64-bit address (EUI-64); otherwise, V := 1 if it is a short 16-bit address.

- F := 0 if the Final Destination Address is an IEEE extended 64-bit address (EUI-64); otherwise, F := 1 if it is a short 16-bit address.

- Hops Left := 0xF (i.e., reserved value indicating that the Deep Hops Left field follows);

- Deep Hops Left := MAX_HOP_LIMIT.

Field definitions of the DFF header are as follows:

Mesh Forw - A 6-bit identifier that allows for the use of different mesh-forwarding mechanisms. As specified in [RFC4944], additional mesh-forwarding mechanisms should use the reserved dispatch byte values following LOWPAN_BC0; therefore, '0 1' MUST precede Mesh Forw. The value of Mesh Forw is LOWPAN_DFF.

DFF fields - A 2-bit version (abbreviated as VER) field; the DUP (abbreviated as D) and RET (abbreviated as R) flags follow after Mesh Forw, as specified in Section 13.2.2. The version specified in this document is '00'. All other bits (besides VER, DUP, and RET) of this octet are reserved and MUST be set to 0.

Sequence Number - A 16-bit field, containing an unsigned integer sequence number, as specified in Section 7.

14. Scope Limitation of DFF

The forwarding mechanism specified in this document MUST be limited in scope to the routing domain in which DFF is used. That also implies that any headers specific to DFF do not traverse the boundaries of the routing domain. This section specifies, both for the "route-over" MoP and the "mesh-under" MoP, how to limit the scope of DFF to the routing domain in which it is used.

Figures 4 to 7 depict four different cases for source and destination of traffic with regards to the scope of the routing domain in which DFF is used. Sections 14.1 and 14.2 specify how routers limit the scope of DFF for the "route-over" MoP and the "mesh-under" MoP, respectively, for these cases. In these sections, all nodes "inside the routing domain" are routers and use DFF, and may also be sources or destinations. Sources or destinations "outside the routing domain" do not run DFF; either they are hosts attached to a router in the routing domain that is running DFF, or they are themselves routers but outside the routing domain and not running DFF.

```
                        +-----------------+
                        |                 |
                        |  (S) ----> (D)  |
                        |                 |
                        +-----------------+
                        Routing Domain
```

Figure 4: Traffic within the Routing Domain (from S to D)

```
                        +-----------------+
                        |                 |
                        |  (S) --------> (R) --------> (D)
                        |                 |
                        +-----------------+
                        Routing Domain
```

Figure 5: Traffic from Within the Routing Domain to Outside of the Domain (from S to D)

```
                        +-----------------+
                        |                 |
         (S) --------> (R) --------> (D)  |
                        |                 |
                        +-----------------+
                        Routing Domain
```

Figure 6: Traffic from Outside the Routing Domain to Inside the Domain (from S to D)

```
                        +-----------------+
                        |                 |
         (S) --------> (R1) -----------> (R2) --------> (D)
                        |                 |
                        +-----------------+
                        Routing Domain
```

Figure 7: Traffic from Outside the Routing Domain, Traversing the Domain and Then to the Outside of the Domain (from S to D)

```
              Key:
                  (S) = source router
                  (D) = destination router
                  (R), (R1), (R2) = other routers
```

14.1. Route-Over MoP

In Figure 4, both the source and destination of the traffic are routers within the routing domain. If traffic is originated at S, the DFF header is added to the IPv6 header (as specified in Section 13.1.2). The Originator Address is set to S and the Destination Address is set to D. The packet is forwarded to D using this specification. When router D receives the packet, it processes the payload of the IPv6 packet in upper layers. This case assumes that S has knowledge that D is in the routing domain, e.g., because of the administrative setting based on the IP address of the destination. If S has no knowledge about whether D is in the routing domain, IPv6-in-IPv6 tunnels as specified in [RFC2473] MUST be used. These cases are described in the following paragraphs.

In Figure 5, the source of the traffic (S) is within the routing domain, and the destination (D) is outside of the routing domain. The IPv6 packet, originated at S, MUST be encapsulated according to [RFC2473] (IPv6-in-IPv6 tunnels) and the DFF header MUST be added to the outer IPv6 header. S chooses the next router that should process the packet as the tunnel exit-point (R). Administrative settings, as well as information from a routing protocol, may be used to determine the tunnel exit-point. If no information is available for which router to choose as the tunnel exit-point, the Next Hop MUST be used as the tunnel exit-point. In some cases, the tunnel exit-point will be the final router along a path towards the packet’s destination, and the packet will only traverse a single tunnel (e.g., if R is a known border router then S can choose R as the tunnel exit-point). In other cases, the tunnel exit-point will not be the final router along the path to D, and the packet may traverse multiple tunnels to reach the destination; note that in this case, the DFF mechanism is only used inside each IPv6-in-IPv6 tunnel. The Originator Address of the packet is set to S and the Destination Address is set to the tunnel exit-point (in the outer IPv6 header). The packet is forwarded to the tunnel exit-point using this specification (potentially using multiple consecutive IPv6-in-IPv6 tunnels). When router R receives the packet, it decapsulates the IPv6 packet and forwards the inner IPv6 packet to D, using normal IPv6 forwarding as specified in [RFC2460].

In Figure 6, the source of the traffic (S) is outside of the routing domain, and the destination (D) is inside of the routing domain. The IPv6 packet, originated at S, is forwarded to R using normal IPv6 forwarding as specified in [RFC2460]. Router R MUST encapsulate the IPv6 packet according to [RFC2473] and add the DFF header (as specified in Section 13.1.2) to the outer IPv6 header. Like in the previous case, R has to select a tunnel exit-point; if it knows that D is in the routing domain (e.g., based on administrative settings), it SHOULD select D as the tunnel exit-point. In case it does not have any information as to which exit-point to select, it MUST use the Next Hop as the tunnel exit-point, limiting the effectiveness of DFF to inside each IPv6-in-IPv6 tunnel. The Originator Address of the packet is set to R, the Destination Address to the tunnel exit-point (both in the outer IPv6 header), and the sequence number in the DFF header is generated locally on R. The packet is forwarded to D using this specification. When router D receives the packet, it decapsulates the inner IPv6 packet and processes the payload of the inner IPv6 packet in upper layers.

This mechanism is typically not used in transit networks; therefore, this case is discouraged, but described nevertheless for completeness. In Figure 7, both the source of the traffic (S) and the destination (D) are outside of the routing domain. The IPv6 packet, originated at S, is forwarded to R1 using normal IPv6 forwarding, as specified in [RFC2460]. Router R1 MUST encapsulate the IPv6 packet according to [RFC2473] and add the DFF header (as specified in Section 13.1.2). R1 selects a tunnel exit-point like in the previous cases; if R2 is, e.g., a known border router, then R1 can select R2 as the tunnel exit-point. The Originator Address is set to R1, the Destination Address is set to the tunnel exit-point (both in the outer IPv6 header), and the sequence number in the DFF header is generated locally on R1. The packet is forwarded to the tunnel exit-point using this specification (potentially traversing multiple consecutive IPv6-in-IPv6 tunnels). When router R2 receives the packet, it decapsulates the inner IPv6 packet and forwards the inner IPv6 packet to D, using normal IPv6 forwarding as specified in [RFC2460].

14.2. Mesh-Under MoP

In Figure 4, both the source and destination of the traffic are routers within the routing domain. If traffic is originated at router S, the LoWPAN-encapsulated packet is created from the IPv6 packet, as specified in [RFC4944]. Then, the Mesh Addressing header and the DFF header (as specified in Section 13.2.2) are added to the LoWPAN encapsulation on router S. The Originator Address is set to S and the Destination Address is set to D. The packet is then forwarded using this specification. When router D receives the packet, it processes the payload of the packet in upper layers.

In Figure 5, the source of the traffic (S) is within the routing domain, and the destination (D) is outside of the routing domain (which is known by S to be outside the routing domain because D uses a different IP prefix from the PAN). The LoWPAN-encapsulated packet, originated at router S, is created from the IPv6 packet as specified in [RFC4944]. Then, the Mesh Addressing header and the DFF header (as specified in Section 13.2.2) are added to the LoWPAN encapsulation on router S. The Originator Address is set to S and the Destination Address is set to R, which is a known border router of the PAN. The packet is then forwarded using this specification. When router R receives the packet, it restores the IPv6 packet from the LoWPAN-encapsulated packet and forwards it to D, using normal IPv6 forwarding, as specified in [RFC2460].

In Figure 6, the source of the traffic (S) is outside of the routing domain, and the destination (D) is inside of the routing domain. The IPv6 packet, originated at S, is forwarded to R using normal IPv6 forwarding, as specified in [RFC2460]. Router R (which is a known border router to the PAN) creates the LoWPAN-encapsulated packet from the IPv6 packet, as specified in [RFC4944]. Then, R adds the Mesh Addressing header and the DFF header (as specified in Section 13.2.2). The Originator Address is set to R, the Destination Address to D, and the sequence number in the DFF header is generated locally on R. The packet is forwarded to D using this specification. When router D receives the packet, it restores the IPv6 packet from the LoWPAN-encapsulated packet and processes the payload in upper layers.

As LoWPANs are typically not transit networks, the following case is discouraged, but described nevertheless for completeness: In Figure 7, both the source of the traffic (S) and the destination (D) are outside of the routing domain. The IPv6 packet, originated at S, is forwarded to R1 using normal IPv6 forwarding, as specified in [RFC2460]. Router R1 (which is a known border router of the PAN) creates the LoWPAN-encapsulated packet from the IPv6 packet, as specified in [RFC4944]. Then, it adds the Mesh Addressing header and the DFF header (as specified in Section 13.2.2). The Originator Address is set to R1, the Destination Address is set to R2 (which is another border router towards the destination), and the sequence number in the DFF header is generated locally on R1. The packet is forwarded to R2 using this specification. When router R2 receives the packet, it restores the IPv6 packet from the LoWPAN-encapsulated packet and forwards the IPv6 packet to D, using normal IPv6 forwarding, as specified in [RFC2460].

15. MTU Exceedance

When adding the DFF header, as specified in Section 9.1, or when encapsulating the packet, as specified in Section 14, the packet size may exceed the MTU. This is described in Section 5 of [RFC2460]. When the packet size of a packet to be forwarded by DFF exceeds the MTU, the following steps apply.

1. The router MUST discard the packet.

2. The router MAY log the event locally (depending on the storage capabilities of the router).

3. The router MUST send back an ICMP "Packet Too Big" message to the source of the packet and report back the Next Hop MTU, which includes the overhead of adding the headers.

16. Security Considerations

Based on the recommendations in [RFC3552], this section describes security threats to DFF and lists which attacks are out of scope, which attacks DFF is susceptible to, and which attacks DFF protects against.

16.1. Attacks That Are Out of Scope

As DFF is a data-forwarding protocol, any security issues concerning the payload of the packets are not considered in this section.

It is the responsibility of upper layers to use appropriate security mechanisms (IPsec, Transport Layer Security (TLS), etc.) according to application requirements. As DFF does not modify the contents of IP datagrams, other than the DFF header (which is a Hop-by-Hop Options extension header in the "route-over" MoP, and therefore not protected by IPsec), no special considerations for IPsec have to be addressed.

Any attack that is not specific to DFF but that applies in general to the link layer (e.g., wireless, Power Line Communication (PLC)) is out of scope. In particular, these attacks are: eavesdropping, packet insertion, packet replay, packet deletion, and man-in-the-middle attacks. Appropriate link-layer encryption can mitigate part of these attacks and is therefore RECOMMENDED.

16.2. Protection Mechanisms of DFF

DFF itself does not provide any additional integrity, confidentiality, or authentication. Therefore, the level of protection of DFF depends on the underlying link-layer security, as well as protection of the payload by upper-layer security (e.g., IPsec).

In the following sections, whenever encrypting or digitally signing packets is suggested for protecting DFF, it is assumed that routers are not compromised.

16.3. Attacks That Are in Scope

This section discusses security threats to DFF, and for each, describes whether (and how) DFF is affected by the threat. DFF is designed to be used in lossy and unreliable networks. Predominant examples of lossy networks are wireless networks, where routers send packets via broadcast. The attacks listed below are easier to exploit in wireless media but can also be observed in wired networks.

16.3.1. Denial of Service

Denial-of-service (DoS) attacks are possible when using DFF by either exceeding the storage on a router or exceeding the available bandwidth of the channel. As DFF does not contain any algorithms with high complexity, it is unlikely that the processing power of the router could be exhausted by an attack on DFF.

The storage of a router can be exhausted by increasing the size of the Processed Set, i.e., by adding new tuples, or by increasing the size of each tuple. New tuples can be added by injecting new packets in the network or by forwarding overheard packets.

Another possible DoS attack is to send packets to a non-existing address in the network. DFF would perform a depth-first search until the Hop Limit has reached zero. It is therefore RECOMMENDED to set the Hop Limit to a value that limits the path length.

If security provided by the link layer is used, this attack can be mitigated if the malicious router does not possess valid credentials, since other routers would not forward data through the malicious router.

16.3.2. Packet Header Modification

The following attacks can be exploited by modifying the packet header information, unless additional security (such as link-layer security) is used.

16.3.2.1. Return Flag Tampering

A malicious router may tamper with the "return" flag of a DFF packet and send it back to the Previous Hop, but only if the malicious router has been selected as the Next Hop by the receiving router (as specified in Section 9.2). If the malicious router had not been selected as the Next Hop, then a returned packet is dropped by the receiving router. Otherwise (i.e., the malicious router had been selected as the Next Hop by the receiving router, and the malicious router has set the return flag), the receiving router then tries alternative neighbors. This may lead to packets never reaching their destination, as well as an unnecessary depth-first search in the network (bandwidth exhaustion / energy drain).

This attack can be mitigated by using appropriate security of the underlying link layer.

16.3.2.2. Duplicate Flag Tampering

A malicious router may modify the Duplicate Flag of a packet that it forwards.

If it changes the flag from 0 to 1, the packet would be detected as a duplicate by other routers in the network and not as a looping packet.

If the Duplicate Flag is changed from 1 to 0, and a router receives that packet for the second time (i.e., it has already received a packet with the same Originator Address and sequence number before), it will wrongly detect a loop.

This attack can be mitigated by using appropriate security of the underlying link layer.

16.3.2.3. Sequence Number Tampering

A malicious router may modify the sequence number of a packet that it forwards.

In particular, if the sequence number is modified to a number of another, previously sent packet of the same Originator, this packet may be wrongly perceived as a looping packet.

This attack can be mitigated by using appropriate security of the underlying link layer.

17. IANA Considerations

IANA has allocated the value 01 000011 for LOWPAN_DFF from the Dispatch Type Field registry.

IANA has allocated the value 0xEE for IP_DFF from the Destination Options and Hop-by-Hop Options registry. The first 3 bits of that value are 111.

18. Acknowledgments

Jari Arkko (Ericsson), Abdussalam Baryun (University of Glamorgan), Antonin Bas (Ecole Polytechnique), Thomas Clausen (Ecole Polytechnifque), Yuichi Igarashi (Hitachi), Kazuya Monden (Hitachi), Geoff Mulligan (Proto6), Hiroki Satoh (Hitachi), Ganesh Venkatesh (Mobelitix), and Jiazi Yi (Ecole Polytechnique) provided useful reviews of the draft and discussions, which helped to improve this document.

The authors also would like to thank Ralph Droms, Adrian Farrel, Stephen Farrell, Ted Lemon, Alvaro Retana, Dan Romascanu, and Martin Stiemerling for their reviews during IETF LC and IESG evaluation.

19. References

19.1. 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.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, December 1998.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

[RFC6564] Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., and M. Bhatia, "A Uniform Format for IPv6 Extension Headers", RFC 6564, April 2012.

[RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.

19.2. Informative References

[DFF_paper1]
Cespedes, S., Cardenas, A., and T. Iwao, "Comparison of Data Forwarding Mechanisms for AMI Networks", 2012 IEEE Innovative Smart Grid Technologies Conference (ISGT), January 2012.

[DFF_paper2]
Iwao, T., Iwao, T., Yura, M., Nakaya, Y., Cardenas, A., Lee, S., and R. Masuoka, "Dynamic Data Forwarding in Wireless Mesh Networks", First IEEE International Conference on Smart Grid Communications (SmartGridComm), October 2010.

[DFS_wikipedia]
Wikipedia, "Depth-first search", May 2013, <http://en.wikipedia.org/w/index.php?title=Depth-first_search&oldid=555203731>.

[KCEC_press_release]
Kit Carson Electric Cooperative (KCEC), "DFF deployed by KCEC", Press Release, 2011, <http://www.kitcarson.com/index.php?option=com_content&view=article&id=45&Itemid=1>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

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

[RFC6775] Shelby, Z., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, November 2012.

Appendix A. Examples

In this section, some example network topologies are depicted, using the DFF mechanism for data forwarding. In these examples, it is assumed there is a routing protocol running that adds or inserts entries into the RIB.

A.1. Example 1: Normal Delivery

Example 1 depicts a network topology with seven routers, A to G, with links between them as indicated by lines. It is assumed that router A sends a packet to G, through B and D, according to the routing protocol.

```
                                        +---+
                                    +---+ D +-----+
                                    |   +---+     |
                            +---+   |             |
                        +---+ B +---+             |
                        |   +---+   |             |
                      +-+-+         |   +---+   +-+-+
                      | A |         +---+ E +---+ G +
                      +-+-+             +---+   +-+-+
                        |   +---+                 |
                        +---+ C +---+             |
                            +---+   |             |
                                    |   +---+     |
                                    +---+ F +-----+
                                        +---+
```

Example 1: Normal Delivery

If no link fails in this topology, and no loop occurs, then DFF forwards the packet along the Next Hops listed in the RIB of each of the routers along the path towards the destination. Each router adds a Processed Tuple for the incoming packet and selects the Next Hop, as specified in Section 11, i.e., it will first select the Next Hop for router G, as determined by the routing protocol.

A.2. Example 2: Forwarding with Link Failure

Example 2 depicts the same topology as Example 1, but both links between B and D and between B and E are unavailable (e.g., because of wireless link characteristics).

```
                                        +---+
                                    XXXX+ D +-----+
                                    X   +---+     |
                            +---+   X             |
                        +---+ B +---+             |
                        |   +---+   X             |
                      +-+-+         X   +---+   +-+-+
                      | A |         XXXX+ E +---+ G +
                      +-+-+             +---+   +-+-+
                        |   +---+                 |
                        +---+ C +---+             |
                            +---+   |             |
                                    |   +---+     |
                                    +---+ F +-----+
                                        +---+
```

Example 2: Link Failure

When B receives the packet from router A, it adds a Processed Tuple and then tries to forward the packet to D. Once B detects that the packet cannot be successfully delivered to D because it does not receive link-layer ACKs, it will follow the procedures listed in Section 10 by setting the DUP flag to 1, selecting E as the new Next Hop, adding E to the list of Next Hops in the Processed Tuple, and then forwarding the packet to E.

As the link to E also fails, B will again follow the procedure in Section 10. As all possible Next Hops (D and E) are listed in the Processed Tuple, B will set the RET flag in the packet and return it to A.

A determines that it already has a Processed Tuple for the returned packet, resets the RET flag of the packet, and selects a new Next Hop for the packet. As B is already in the list of Next Hops in the Processed Tuple, it will select C as the Next Hop and forward the packet to it. C will then forward the packet to F, and F delivers the packet to its destination G.

A.3. Example 3: Forwarding with Missed Link-Layer Acknowledgment

Example 3 depicts the same topology as Example 1, but the link-layer acknowledgments from C to A are lost (e.g., because the link is unidirectional). It is assumed that A prefers a path to G through C and F.

```
                                        +---+
                                    +---+ D +-----+
                                    |   +---+     |
                            +---+   |             |
                        +---+ B +---+             |
                        |   +---+   |             |
                      +-+-+         |   +---+   +-+-+
                      | A |         +---+ E +---+ G +
                      +-+-+             +---+   +-+-+
                        .   +---+                 |
                        +...+ C +---+             |
                            +---+   |             |
                                    |   +---+     |
                                    +---+ F +-----+
                                        +---+
```

Example 3: Missed Link-Layer Acknowledgment

While C successfully receives the packet from A, A does not receive the L2 ACK and assumes the packet has not been delivered to C. Therefore, it sets the DUP flag of the packet to 1, in order to indicate that this packet may be a duplicate. Then, it forwards the packet to B.

A.4. Example 4: Forwarding with a Loop

Example 4 depicts the same topology as Example 1, but there is a loop from D to A, and A sends the packet to G through B and D.

```
                        +-----------------+
                        |                 |
                        |               +-+-+
                        |           +---+ D +
                        |           |   +---+
                       \|/  +---+   |
                        +---+ B +---+
                        |   +---+   |
                      +-+-+         |   +---+   +-+-+
                      | A |         +---+ E +---+ G +
                      +-+-+             +---+   +-+-+
                        |   +---+                 |
                        +---+ C +---+             |
                            +---+   |             |
                                    |   +---+     |
                                    +---+ F +-----+
                                        +---+
```

Example 4: Loop

When A receives the packet through the loop from D, it will find a Processed Tuple for the packet. Router A will set the RET flag and return the packet to D, which in turn will return it to B. B will then select E as the Next Hop, which will then forward it to G.

Appendix B. Deployment Experience

DFF has been deployed and experimented with both in real deployments and in network simulations, as described below.

B.1. Deployments in Japan

The majority of the large Advanced Metering Infrastructure (AMI) deployments using DFF are located in Japan, but the data of these networks is the property of Japanese utilities and cannot be disclosed.

B.2. Kit Carson Electric Cooperative

DFF has been deployed at Kit Carson Electric Cooperative (KCEC), a non-profit organization distributing electricity to about 30,000 customers in New Mexico. As described in a press release [KCEC_press_release], DFF is running on currently about 2000 electric meters. All meters are connected through a mesh network using an unreliable, wireless medium. DFF is used together with a distance-vector routing protocol. Metering data from each meter is sent towards a gateway periodically (every 15 minutes). The data delivery reliability is over 99%.

B.3. Simulations

DFF has been evaluated in Ns2 (http://nsnam.isi.edu/nsnam) and OMNEST (http://www.omnest.com) simulations, in conjuction with a distance-vector routing protocol. The performance of DFF has been compared to using only the routing protocol without DFF. The results published in peer-reviewed academic papers [DFF_paper1] [DFF_paper2] show significant improvements of the packet delivery ratio compared to using only the distance-vector protocol.

B.4. Open-Source Implementation

Fujitsu Laboratories of America is currently working on an open-source implementation of DFF, which will be released in 2013 and will allow for interoperability testings of different DFF implementations. The implementation is written in Java and can be used both on real machines and in the Ns2 simulator.

Authors’ Addresses

Ulrich Herberg (editor)
Fujitsu
1240 E. Arques Avenue, M/S 345
Sunnyvale, CA 94085
USA
Phone: +1 408 530 4528
EMail: ulrich.herberg@us.fujitsu.com

Alvaro A. Cardenas
University of Texas at Dallas
School of Computer Science, 800 West Campbell Rd, EC 31
Richardson, TX 75080-3021
USA
EMail: alvaro.cardenas@me.com

Tadashige Iwao
Fujitsu
Shiodome City Center, 5-2, Higashi-shimbashi 1-chome, Minato-ku
Tokyo,
JP
Phone: +81-44-754-3343
EMail: smartnetpro-iwao_std@ml.css.fujitsu.com

Michael L. Dow
Freescale
6501 William Cannon Drive West
Austin, TX 78735
USA
Phone: +1 512 895 4944
EMail: m.dow@freescale.com

Sandra L. Cespedes
Icesi University
Calle 18 #122-135, Pance
Cali,
Colombia
Phone: +57 (2) 5552334
EMail: scespedes@icesi.edu.co