            Multicast using Bit Index Explicit Replication
                  draft-ietf-bier-architecture-01

Abstract

   This document specifies a new architecture for the forwarding of
   multicast data packets. It provides optimal forwarding of multicast
   packets through a "multicast domain". However, it does not require a
   protocol for explicitly building multicast distribution trees, nor
   does it require intermediate nodes to maintain any per-flow state.
   This architecture is known as "Bit Index Explicit Replication"
   (BIER). When a multicast data packet enters the domain, the ingress
   router determines the set of egress routers to which the packet needs
   to be sent. The ingress router then encapsulates the packet in a
   BIER header. The BIER header contains a bitstring in which each bit
   represents exactly one egress router in the domain; to forward the
   packet to a given set of egress routers, the bits corresponding to
   those routers are set in the BIER header. Elimination of the per-
   flow state and the explicit tree-building protocols results in a
   considerable simplification.

Status of This Memo

   This Internet-Draft will expire on December 27, 2015.

Copyright Notice

Table of Contents

1. Introduction

   This document specifies a new architecture for the forwarding of
   multicast data packets. It provides optimal forwarding of multicast
   data packets through a "multicast domain". However, it does not
   require the use of a protocol for explicitly building multicast
   distribution trees, and it does not require intermediate nodes to
   maintain any per-flow state. This architecture is known as "Bit
   Index Explicit Replication" (BIER).

   A router that supports BIER is known as a "Bit-Forwarding Router"
   (BFR). The BIER control plane protocols (see Section 4.2) run within
   a "BIER domain", allowing the BFRs within that domain to exchange the
   information needed for them to forward packets to each other using
   BIER.

   A multicast data packet enters a BIER domain at a "Bit-Forwarding
   Ingress Router" (BFIR), and leaves the BIER domain at one or more
   "Bit-Forwarding Egress Routers" (BFERs). A BFR that receives a
   multicast data packet from another BFR in the same BIER domain, and
   forwards the packet to another BFR in the same BIER domain, will be
   known as a "transit BFR" for that packet. A single BFR may be a BFIR
   for some multicast traffic while also being a BFER for some multicast
   traffic and a transit BFR for some multicast traffic. In fact, a BFR
   may be the BFIR for a given packet and may also be (one of) the
   BFER(s), for that packet; it may also forward that packet to one or
   more additional BFRs.

   A BIER domain may contain one or more sub-domains. Each BIER domain
   MUST contain at least one sub-domain, the "default sub-domain" (also
   denoted "sub-domain zero"). If a BIER domain contains more than one
   sub-domain, each BFR in the domain MUST be provisioned to know the
   set of sub-domains to which it belongs. Each sub-domain is
   identified by a sub-domain-id in the range [0,255].

   For each sub-domain to which a given BFR belongs, if the BFR is
   capable of acting as a BFIR or a BFER, it MUST be provisioned with a
   "BFR-id" that is unique within the sub-domain. A BFR-id is a small
   unstructured positive integer. For instance, if a particular BIER
   sub-domain contains 1,374 BFRs, each one could be given a BFR-id in
   the range 1-1374.

If a given BFR belongs to more than one sub-domain, it may (though it need not) have a different BFR-id for each sub-domain.

When a multicast packet arrives from outside the domain at a BFIR, the BFIR determines the set of BFERs to which the packet will be sent. The BFIR also determines the sub-domain in which the packet will be sent. Determining the sub-domain in which a given packet will be sent is known as "assigning the packet to a sub-domain". Procedures for choosing the sub-domain to which a particular packet is assigned are outside the scope of this document. However, once a particular packet has been assigned to a particular sub-domain, it remains assigned to that sub-domain until it leaves the BIER domain. That is, the sub-domain to which a packet is assigned MUST NOT be changed while the packet is in flight through the BIER domain.

Once the BFIR determines sub-domain and the set of BFERs for a given packet, the BFIR encapsulates the packet in a "BIER header". The BIER header contains a bit string in which each bit represents a single BFR-id. To indicate that a particular BFER is to receive a given packet, the BFIR sets the bit corresponding to that BFER's BFR-id in the sub-domain to which the packet has been assigned. We will use term "BitString" to refer to the bit string field in the BIER header. We will use the term "payload" to refer to the packet that has been encapsulated. Thus a "BIER-encapsulated" packet consists of a "BIER header" followed by a "payload".

The number of BFERs to which a given packet can be forwarded is limited only by the length of the BitString in the BIER header. Different deployments can use different BitString lengths. We will use the term "BitStringLength" to refer to the number of bits in the BitString. It is possible that some deployment will have more BFERs in a given sub-domain than there are bits in the BitString. To accommodate this case, the BIER encapsulation includes both the BitString and a "Set Identifier" (SI). It is the BitString and the SI together that determine the set of BFERs to which a given packet will be delivered:

o by convention, the least significant (rightmost) bit in the BitString is "bit 1", and the most significant (leftmost) bit is "bit BitStringLength".

o if a BIER-encapsulated packet has an SI of n, and a BitString with bit k set, then the packet must be delivered to the BFER whose BFR-id (in the sub-domain to which the packet has been assigned) is n*BitStringLength+k.

For example, suppose the BIER encapsulation uses a BitStringLength of 256 bits. By convention, the least significant (rightmost) bit is

"bit 1", and the most significant (leftmost) bit is "bit 256".
Suppose that a given packet has been assigned to sub-domain 0, and
needs to be delivered to three BFERs, where those BFERs have BFR-ids
in sub-domain 0 of 13, 126, and 235 respectively. The BFIR would
create a BIER encapsulation with the SI set to zero, and with bits
13, 126, and 235 of the BitString set. (All other bits of the
BitString would be clear.) If the packet also needs to be sent to a
BFER whose BFR-id is 257, the BFIR would have to create a second copy
of the packet, and the BIER encapsulation would specify an SI of 1,
and a BitString with bit 1 set and all the other bits clear.

Note that it is generally advantageous to assign the BFR-ids of a
given sub-domain so that as many BFERs as possible can be represented
in a single bit string.

Suppose a BFR, call it BFR-A, receives a packet whose BIER
encapsulation specifies an SI of 0, and a BitString with bits 13, 26,
and 235 set. Suppose BFR-A has two BFR neighbors, BFR-B and BFR-C,
such that the best path to BFERs 13 and 26 is via BFR-B, but the best
path to BFER 235 is via BFR-C. Then BFR-A will replicate the packet,
sending one copy to BFR-B and one copy to BFR-C. However, BFR-A will
clear bit 235 in the BitString of the packet copy it sends to BFR-B,
and will clear bits 13 and 26 in the BitString of the packet copy it
sends to BFR-C. As a result, BFR-B will forward the packet only
towards BFERs 13 and 26, and BFR-C will forward the packet only
towards BFER 235. This ensures that each BFER receives only one copy
of the packet.

With this forwarding procedure, a multicast data packet can follow an
optimal path from its BFIR to each of its BFERs. Further, since the
set of BFERs for a given packet is explicitly encoded into the BIER
header, the packet is not sent to any BFER that does not need to
receive it. This allows for optimal forwarding of multicast traffic.
This optimal forwarding is achieved without any need for transit BFRs
to maintain per-flow state, or to run a multicast tree-building
protocol.

The idea of encoding the set of egress nodes into the header of a
multicast packet is not new. For example, [Boivie_Feldman] proposes
to encode the set of egress nodes as a set of IP addresses, and
proposes mechanisms and procedures that are in some ways similar to
those described in the current document. However, since BIER encodes
each BFR-id as a single bit in a bit string, it can represent up to
128 BFERs in the same number of bits that it would take to carry the
IPv6 address of a single BFER. Thus BIER scales to a much larger
number of egress nodes per packet.

BIER does not require that each transit BFR look up the best path to
each BFER that is identified in the BIER header; the number of
lookups required in the forwarding path for a single packet can be
limited to the number of neighboring BFRs; this can be much smaller
than the number of BFERs. See Section 6 (especially Section 6.4) for
details.

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].

2. The BFR Identifier and BFR-Prefix

Each BFR MUST be assigned a "BFR-Prefix". A BFR's BFR-Prefix MUST be
an IP address (either IPv4 or IPv6) of the BFR, and MUST be unique
and routable within the BIER domain. It is RECOMMENDED that the
BFR-prefix be a loopback address of the BFR. Two BFRs in the same
BIER domain MUST NOT be assigned the same BFR-Prefix. Note that a
BFR in a given BIER domain has the same BFR-prefix in all the sub-
domains of that BIER domain.

A "BFR Identifier" (BFR-id) is a number in the range [1,65535]. In
general, each BFR in a given BIER sub-domain must be assigned a
unique number from this range (i.e., two BFRs in the same BIER sub-
domain MUST NOT have the same BFR-id in that sub-domain). However,
if it is known that a given BFR will never need to function as a BFER
in a given sub-domain, then it is not necessary to assign a BFR-id
for that sub-domain to that BFR.

Note that the value 0 is not a legal BFR-id.

The procedure for assigning a particular BFR-id to a particular BFR
is outside the scope of this document. However, it is RECOMMENDED
that the BFR-ids for each sub-domain be assigned "densely" from the
numbering space, as this will result in a more efficient encoding
(see Section 3). That is, if there are 256 or fewer BFERs, it is
RECOMMENDED to assign all the BFR-ids from the range [1,256]. If
there are more than 256 BFERs, but less than 512, it is RECOMMENDED
to assign all the BFR-ids from the range [1,512], with as few "holes"
as possible in the earlier range. However, in some deployments, it
may be advantageous to depart from this recommendation; this is
discussed further in Section 3.

3. Encoding BFR Identifiers in BitStrings

To encode a BFR-id in a BIER data packet, one must convert the BFR-id
to an SI and a BitString. This conversion depends upon the parameter

we are calling "BitStringLength". The conversion is done as follows.
If the BFR-id is N, then

o SI is the integer part of the quotient (N-1)/BitStringLength

o The BitString has one bit position set. If the low-order bit is
   bit 1, and the high-order bit is bit BitStringLength, the bit
   position that represents BFR-id N is
   ((N-1) modulo BitStringLength)+1.

If several different BFR-ids all resolve to the same SI, then all
those BFR-ids can be represented in a single BitString. The
BitStrings for all of those BFR-ids are combined using a bitwise
logical OR operation.

Different BIER domains may use different values of BitStringLength.
Each BFR in a given BIER domain MUST be provisioned to know the
following:

o the BitStringLength ("Imposition BitStringLength") and sub-domain
   ("Imposition sub-domain") to use when it imposes (as a BFIR) a
   BIER encapsulation on a particular set of packets, and

o the BitStringLengths ("Disposition BitStringLengths") that it will
   process when (as a BFR or BFER) it receives packets from a
   particular sub-domain.

It is not required that a BFIR use the same Imposition
BitStringLength or the same Imposition sub-domain for all packets on
which it imposes the BIER encapsulation. However, if a particular
BFIR is provisioned to use a particular Imposition BitStringLength
and a particular Imposition sub-domain when imposing the
encapsulation on a given set of packets, all other BFRs with BFR-ids
in that sub-domain SHOULD be provisioned to process received BIER
packets with that BitStringLength (i.e., all other BFRs with BFR-ids
in that sub-domain SHOULD be provisioned with that BitStringLength as
a Disposition BitStringLength for that sub-domain. Exceptions to
this rule MAY be made under certain conditions; this is discussed in
Section 6.10.

Every BFIR MUST be capable of being provisioned with an Imposition
BitStringLength of 256. Every BFR and BFER MUST be capable of being
provisioned with a Disposition BitStringLength of 256.

Particular BIER encapsulation types MAY allow other BitStringLengths
to be OPTIONALLY supported. For example, when using the
encapsulation specified in [MPLS_BIER_ENCAPS], a BFR may be capable
of being provisioned to use any or all of the following

BitStringLengths as Imposition BitStringLengths and as Disposition
BitStringLengths: 64, 128, 256, 512, 1024, 2048, and 4096.

If a BFR is capable of being provisioned with a given value of
BitStringLength as an Imposition BitStringLength, it MUST also be
capable of being provisioned with that same value as one of its
Disposition BitStringLengths. It SHOULD be capable of being
provisioned with all legal smaller values of BitStringLength as both
Imposition and Disposition BitStringLength.

In order to support transition from one BitStringLength to another,
every BFR MUST be capable of being provisioned to simultaneously use
two different Disposition BitStringLengths.

A BFR MUST support SI values in the range [0,15], and MAY support SI
values in the range [0,255]. ("Supporting the values in a given
range" means, in this context, that any value in the given range is
legal, and will be properly interpreted.)

When a BFIR determines that a multicast data packet, assigned to a
given sub-domain, needs to be forwarded to a particular set of
destination BFERs, the BFIR partitions that set of BFERs into
subsets, where each subset contains the target BFERs whose BFR-ids in
the given sub-domain all resolve to the same SI. Call these the
"SI-subsets" for the packet. Each SI-subset can be represented by a
single BitString. The BFIR creates a copy of the packet for each
SI-subset. The BIER encapsulation is then applied to each packet.
The encapsulation specifies a single SI for each packet, and contains
the BitString that represents all the BFR-ids in the corresponding
SI-subset. Of course, in order to properly interpret the BitString,
it must be possible to infer the sub-domain-id from the encapsulation
as well.

Suppose, for example, that a BFIR determines that a given packet
needs to be forwarded to three BFERs, whose BFR-ids (in the
appropriate sub-domain) are 27, 235, and 497. The BFIR will have to
forward two copies of the packet. One copy, associated with SI=0,
will have a BitString with bits 27 and 235 set. The other copy,
associated with SI=1, will have a BitString with bit 241 set.

In order to minimize the number of copies that must be made of a
given multicast packet, it is RECOMMENDED that the BFR-ids be
assigned "densely" (see Section 2) from the numbering space. This
will minimize the number of SIs that have to be used in the domain.
However, depending upon the details of a particular deployment, other
assignment methods may be more advantageous. Suppose, for example,
that in a certain deployment, every multicast flow is either intended
for the "east coast" or for the "west coast". In such a deployment,

it would be advantageous to assign BFR-ids so that all the "west
coast" BFR-ids fall into the same SI-subset, and so that all the
"east coast" BFR-ids fall into the same SI-subset.

When a BFR receives a BIER data packet, it will infer the SI from the
encapsulation. The set of BFERs to which the packet needs to be
forwarded can then be inferred from the SI and the BitString.

In some of the examples given later in this document, we will use a
BitStringLength of 4, and will represent a BFR-id in the form
"SI:xyzw", where SI is the Set Identifier of the BFR-id (assuming a
BitStringLength of 4), and xyzw is a string of 4 bits. A
BitStringLength of 4 is used only in the examples; we would not
expect actual deployments to have such a small BitStringLength.

It is possible that several different forms of BIER encapsulation
will be developed. If so, the particular encapsulation that is used
in a given deployment will depend on the type of network
infrastructure that is used to realize the BIER domain. Details of
the BIER encapsulation(s) will be given in companion documents. An
encapsulation for use in MPLS networks is described in
[MPLS_BIER_ENCAPS]

4. Layering

It is helpful to think of the BIER architecture as consisting of
three layers: the "routing underlay", the "BIER layer", and the
"multicast flow overlay".

4.1. The Routing Underlay

The "routing underlay" establishes "adjacencies" between pairs of
BFRs, and determines one or more "best paths" from a given BFR to a
given set of BFRs. Each such path is a sequence of BFRs <BFR(k),
BFR(k+1), ..., BFR(k+n)> such that BFR(k+j) is "adjacent" to
BFR(k+j+1) (for 0<=j<n).

At a given BFR, say BFR-A, for every IP address that is the address
of a BFR in the BIER domain, the routing underlay will map that IP
address into a set of one or more "equal cost" adjacencies. If a
BIER data packet has to be forwarded by BFR-A to a given BFER, say
BFER-B, the packet will follow the path from BFR-A to BFER-B that is
determined by the routing underlay.

It is expected that in a typical deployment, the routing underlay
will be the default topology that the Interior Gateway Protocol
(IGP), e.g., OSPF, uses for unicast routing. In that case, the
underlay adjacencies are just the OSPF adjacencies. A BIER data

packet traveling from BFR-A to BFER-B will follow the path that OSPF
has selected for unicast traffic from BFR-A to BFER-B.

If one wants to have multicast traffic from BFR-A to BFER-B travel a
path that is different from the path used by the unicast traffic from
A to B, one can use a different underlay. For example, if multi-
topology OSPF is being used, one OSPF topology could be used for
unicast traffic, and the other for multicast traffic. (Each topology
would be considered to be a different underlay.) Alternatively, one
could deploy a routing underlay that creates a multicast-specific
tree of some sort, perhaps a Steiner tree. Then BIER could be used
to forward multicast data packets along the multicast-specific tree,
while unicast packets follow the "ordinary" OSPF best path. It is
even possible to have multiple routing underlays used by BIER, as
long as one can infer from a data packet's BIER encapsulation which
underlay is being used for that packet.

If multiple routing underlays are used in a single BIER domain, each
BIER sub-domain MUST be associated with a single routing underlay.
(Though multiple sub-domains may be associated with the same routing
underlay.) A BFR that belongs to multiple sub-domains MUST be
provisioned to know which routing underlay is used by each sub-
domain. By default (i.e., in the absence of any provisioning to the
contrary), each sub-domain uses the default topology of the unicast
IGP as the routing underlay.

Note that specification of the protocol and procedures of the routing
underlay is outside the scope of this document.

4.2. The BIER Layer

The BIER layer consists of the protocol and procedures that are used
in order to transmit a multicast data packet across a BIER domain,
from its BFIR to its BFERs. This includes the following components:

o Protocols and procedures that a given BFR uses to advertise, to
   all other BFRs in the same BIER domain:

   * its BFR-prefix;

   * its BFR-id in each sub-domain for which it has been provisioned
      with a BFR-id;

   * the set of Disposition BitStringLengths it has been provisioned
      to use for each sub-domain;

   * optionally, information about the routing underlay associated
      with each sub-domain.

o The procedures used by a BFIR to impose a BIER header on a
   multicast data packet.

o The procedures for forwarding BIER-encapsulated packets, and for
   modifying the BIER header during transit.

4.3. The Multicast Flow Overlay

The "multicast flow overlay" consists of the set of protocols and
procedures that enable the following set of functions.

o When a BFIR receives a multicast data packet from outside the BIER
   domain, the BFIR must determine the set of BFERs for that packet.
   This information is provided by the multicast flow overlay.

o When a BFER receives a BIER-encapsulated packet from inside the
   BIER domain, the BFER must determine how to further forward the
   packet. This information is provided by the multicast flow
   overlay.

For example, suppose the BFIR and BFERs are Provider Edge (PE)
routers providing Multicast Virtual Private Network (MVPN) service.
The multicast flow overlay consists of the protocols and procedures
described in [RFC6513] and [RFC6514]. The MVPN signaling described
in those RFCs enables an ingress PE to determine the set of egress
PEs for a given multicast flow (or set of flows); it also enables an
egress PE to determine the "Virtual Routing and Forwarding Tables"
(VRFs) to which multicast packets from the backbone network should be
sent. MVPN signaling also has several components that depend on the
type of "tunneling technology" used to carry multicast data though
the network. Since BIER is, in effect, a new type of "tunneling
technology", some extensions to the MVPN signaling are needed in
order to properly interface the multicast flow overlay with the BIER
layer. These will be specified in a companion document.

MVPN is just one example of a multicast flow overlay. Protocols and
procedures for other overlays will be provided in companion
documents. It is also possible to implement the multicast flow
overlay by means of a "Software Defined Network" (SDN) controller.
Specification of the protocols and procedures of the multicast flow
overlay is outside the scope of this document.

5. Advertising BFR-ids and BFR-Prefixes

As stated in Section 2, each BFER is assigned (by provisioning) a
BFR-id (for a given BIER sub-domain). Each BFER must advertise these
assignments to all the other BFRs in the domain. Similarly, each BFR
is assigned (by provisioning) a BFR-prefix (for a given BIER domain),

and must advertise this assignment to all the other BFRs in the
domain. Finally, each BFR has been provisioned to use a certain set
of Disposition BitStringLengths for each sub-domain, and must
advertise these to all other BFRs in the domain.

If the BIER domain is also a link state routing IGP domain (i.e., an
OSPF or IS-IS domain), the advertisement of the BFR-prefix, <sub-
domain-id,BFR-id> and BitStringLength can be done using the
advertisement capabilities of the IGP. For example, if a BIER domain
is also an OSPF domain, these advertisements can be done using the
OSPF "Opaque Link State Advertisement" (Opaque LSA) mechanism.
Details of the necessary extensions to OSPF and IS-IS will be
provided in companion documents. (See [OSPF_BIER_EXTENSIONS] and
[ISIS_BIER_EXTENSIONS].)

These advertisements enable each BFR to associate a given <sub-
domain-id, BFR-id> with a given BFR-prefix. As will be seen in
subsequent sections of this document, knowledge of this association
is an important part of the forwarding process.

Since each BFR needs to have a unique (in each sub-domain) BFR-id,
two different BFRs will not advertise ownership of the same <sub-
domain-id, BFR-id> unless there has been a provisioning error.

o If BFR-A determines that BFR-B and BFR-C have both advertised the
   same BFR-id for the same sub-domain, BFR-A MUST log an error.
   Suppose that the duplicate BFR-id is "N". When BFR-A is
   functioning as a BFIR, it MUST NOT encode the BFR-id value N in
   the BIER encapsulation of any packet that has been assigned to the
   given sub-domain, even if it has determined that the packet needs
   to be received by BFR-B and/or BFR-C.

   This will mean that BFR-B and BFR-C cannot receive multicast
   traffic at all in the given sub-domain until the provisioning
   error is fixed. However, that is preferable to having them
   receive each other's traffic.

o If BFR-A has been provisioned with BFR-id N for a particular sub-
   domain, has not yet advertised its ownership of BFR-id N for that
   sub-domain, but has received an advertisement from a different BFR
   (say BFR-B) that is advertising ownership of BFR-id N for the same
   sub-domain, then BFR-A SHOULD log an error, and MUST NOT advertise
   its own ownership of BFR-id N for that sub-domain as long as the
   advertisement from BFR-B is extant.

   This procedure may prevent the accidental misconfiguration of a
   new BFR from impacting an existing BFR.

If a BFR advertises that it has a BFR-id of 0 in a particular sub-
domain, other BFRs receiving the advertisement MUST interpret that
advertisement as meaning that the advertising BFR does not have a
BFR-id in that sub-domain.

6. BIER Intra-Domain Forwarding Procedures

This section specifies the rules for forwarding a BIER-encapsulated
data packet within a BIER domain.

6.1. Overview

This section provides a brief overview of the BIER forwarding
procedures. Subsequent sub-sections specify the procedures in more
detail.

To forward a BIER-encapsulated packet:

1. Determine the packet's sub-domain.

2. Determine the packet's BitStringLength and BitString.

3. Determine the packet's SI.

4. From the sub-domain, the SI and the BitString, determine the set
    of destination BFERs for the packet.

5. Using information provided by the routing underlay associated
    with the packet's sub-domain, determine the next hop adjacency
    for each of the destination BFERs.

6. Partition the set of destination BFERs such that all the BFERs in
    a single partition have the same next hop. We will say that each
    partition is associated with a next hop.

7. For each partition:

    a. Make a copy of the packet.

    b. Clear any bit in the packet's BitString that identifies a
        BFER that is not in the partition.

    c. Transmit the packet to the associated next hop.

If a BFR receives a BIER-encapsulated packet whose sub-domain, SI and
BitString identify that BFR itself, then the BFR is also a BFER for
that packet. As a BFER, it must pass the payload to the multicast
flow overlay. If the BitString has bits set for other BFRs, the

packet also needs to be forwarded further within the BIER domain. If
the BF(E)R also forwards one or more copies of the packet within the
BIER domain, the bit representing the BFR's own BFR-id MUST be clear
in all the copies.

When BIER on a BFER passes a packet to the multicast flow overlay, it
may need to provide contextual information obtained from the BIER
encapsulation. The information that needs to pass between the BIER
layer and the multicast flow overlay is specific to the multicast
flow overlay. Specification of the interaction between the BIER
layer and the multicast flow overlay is outside the scope of this
specification.

When BIER on a BFER passes a packet to the multicast flow overlay,
the overlay will determine how to further dispatch the packet. If
the packet needs to be forwarded into another BIER domain, then the
BFR will act as a BFER in one BIER domain and as a BFIR in another.
A BIER-encapsulated packet cannot pass directly from one BIER domain
to another; at the boundary between BIER domains, the packet must be
decapsulated and passed to the multicast flow overlay.

Note that when a BFR transmits multiple copies of a packet within a
BIER domain, only one copy will be destined to any given BFER.
Therefore it is not possible for any BIER-encapsulated packet to be
delivered more than once to any BFER.

6.2. BFR Neighbors

The "BFR Neighbors" (BFR-NBRs) of a given BFR, say BFR-A, are those
BFRs that, according to the routing underlay, are adjacencies of
BFR-A. Each BFR-NBR will have a BFR-prefix.

Suppose a BIER-encapsulated packet arrives at BFR-A. From the
packet's encapsulation, BFR-A learns the sub-domain of the packet,
and the BFR-ids (in that sub-domain) of the BFERs to which the packet
is destined. Then using the information advertised per Section 5,
BFR-A can find the BFR-prefix of each destination BFER. Given the
BFR-prefix of a particular destination BFER, say BFER-D, BFR-A learns
from the routing underlay (associated with the packet's sub-domain)
an IP address of the BFR that is the next hop on the path from BFR-A
to BFER-D. Let's call this next hop BFR-B. BFR-A must then
determine the BFR-prefix of BFR-B. (This determination can be made
from the information advertised per Section 5.) This BFR-prefix is
the BFR-NBR of BFR-A on the path from BFR-A to BFER-D.

Note that if the routing underlay provides multiple equal cost paths
from BFR-A to BFER-D, BFR-A may have multiple BFR-NBRs for BFER-D.

Under certain circumstances, a BFR may have adjacencies (in a
particular routing underlay) that are not BFRs. Please see
Section 6.9 for a discussion of how to handle those circumstances.

6.3. The Bit Index Routing Table

The Bit Index Routing Table (BIRT) is a table that maps from the
BFR-id (in a particular sub-domain) of a BFER to the BFR-prefix of
that BFER, and to the BFR-NBR on the path to that BFER.

```
   ( A ) ----------- (  B  ) ----------- ( C ) ----------- ( D )
  4 (0:1000)              \                  \             1 (0:0001)
                           \                  \
                          ( E )              ( F )
                        3 (0:0100)        2 (0:0010)
```
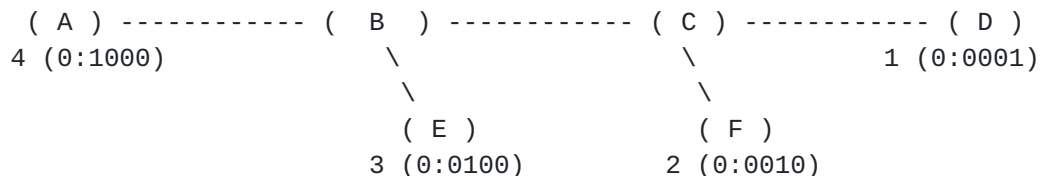
Figure 1: BIER Topology 1

As an example, consider the topology shown in Figure 1. In this
diagram, we represent the BFR-id of each BFR in the SI:xyzw form
discussed in Section 3. This topology will result in the BIRT of
Figure 2 at BFR-B. The first column shows the BFR-id as a number and
also (in parentheses) in the SI:BitString format that corresponds to
a BitStringLength of 4. (The actual minimum BitStringLength is 64,
but we use 4 in the examples.)

Note that a BIRT is specific to a particular BIER sub-domain.

```
-----------------------------------------------
|     BFR-id    |  BFR-Prefix  | BFR-NBR  |
| (SI:BitString) | of Dest BFER |          |
===============================================
|   4 (0:1000)  |      A       |    A     |
-----------------------------------------------
|   1 (0:0001)  |      D       |    C     |
-----------------------------------------------
|   3 (0:0100)  |      E       |    E     |
-----------------------------------------------
|   2 (0:0010)  |      F       |    C     |
-----------------------------------------------
```

Figure 2: Bit Index Routing Table at BFR-B

6.4. The Bit Index Forwarding Table

   The "Bit Index Forwarding Table" (BIFT) is derived from the BIRT as
   follows. (Note that a BIFT is specific to a particular sub-domain.)

   Suppose that several rows in the BIRT have the same SI and the same
   BFR-NBR. By taking the logical OR of the BitStrings of those rows,
   we obtain a bit mask that corresponds to that combination of SI and
   BFR-NBR. We will refer to this bit mask as the "Forwarding Bit Mask"
   (F-BM) for that <SI,BFR-NBR> combination.

   For example, in Figure 2, we see that two of the rows have the same
   SI (0) and same BFR-NBR (C). The Bit Mask that corresponds to <SI=0,
   BFR-NBR-C> is 0011 ("0001" OR'd with "0010").

   The BIFT is used to map from the BFR-id of a BFER to the
   corresponding F-BM and BFR-NBR. For example, Figure 3 shows the BIFT
   that is derived from the BIRT of Figure 2. Note that BFR-ids 1 and 2
   have the same SI and the same BFR-NBR, hence they have the same F-BM.

```
            ---------------------------------------
            |      BFR-id   |  F-BM  | BFR-NBR |
            | (SI:Bitstring) |        |         |
            =======================================
            |   1 (0:0001)  |  0011  |    C    |
            ---------------------------------------
            |   2 (0:0010)  |  0011  |    C    |
            ---------------------------------------
            |   3 (0:0100)  |  0100  |    E    |
            ---------------------------------------
            |   4 (0:1000)  |  1000  |    A    |
            ---------------------------------------
```

                Figure 3: Bit Index Forwarding Table

   This Bit Index Forwarding Table (BIFT) is programmed into the data-
   plane and used to forward packets, applying the rules specified below
   in Section 6.5.

6.5. The BIER Forwarding Procedure

   Below is the procedure that a BFR uses for forwarding a BIER-
   encapsulated packet.

   1. Determine the packet's SI, BitStringLength, and sub-domain.

2. If the BitString consists entirely of zeroes, discard the packet;
   the forwarding process has been completed. Otherwise proceed to
   step 3.

3. Find the position, call it "k", of the least significant (i.e.,
   of the rightmost) bit that is set in the packet's BitString.
   (Remember, bits are numbered from 1, starting with the least
   significant bit.)

4. If bit k identifies the BFR itself, copy the packet, and send the
   copy to the multicast flow overlay. Then clear bit k in the
   original packet, and go to step 2. Otherwise, proceed to step 5.

5. Use the value k, together with the SI, sub-domain, and
   BitStringLength, as the 'index' into the BIFT.

6. Extract from the BIFT the F-BM and the BFR-NBR.

7. Copy the packet. Update the copy's BitString by AND'ing it with
   the F-BM (i.e., PacketCopy->BitString &= F-BM). Then forward the
   copy to the BFR-NBR. Note that when a packet is forwarded to a
   particular BFR-NBR, its BitString identifies only those BFERs
   that are to be reached via that BFR-NBR.

8. Now update the original packet's BitString by AND'ing it with the
   INVERSE of the F-BM (i.e., Packet->Bitstring &= ~F-BM). (This
   clears the bits that identify the BFERs to which a copy of the
   packet has just been forwarded.) Go to step 2.

Note that this procedure causes the packet to be forwarded to a
particular BFR-NBR only once. The number of lookups in the BIFT is
the same as the number of BFR-NBRs to which the packet must be
forwarded; it is not necessary to do a separate lookup for each
destination BFER.

Suppose it has been decided (by the above rules) to send a packet to
a particular BFR-NBR. If that BFR-NBR is connected via multiple
parallel interfaces, it may be desirable to apply some form of load
balancing. Load balancing algorithms are outside the scope of this
document. However, if the packet's encapsulation contains an
"entropy" field, the entropy field SHOULD be respected; two packets
with the same value of the entropy field SHOULD be sent on the same
interface (if possible).

In some cases, the routing underlay may provide multiple equal cost
paths (through different BFR-NBRs) to a given BFER. This is known as
"Equal Cost Multiple Paths" (ECMP). The procedures described in this

section must be augmented in order to support load balancing over
ECMP. The necessary augmentations can be found in Section 6.7.

In the event that unicast traffic to the BFR-NBR is being sent via a
"bypass tunnel" of some sort, the BIER-encapsulated multicast traffic
send to the BFR-NBR SHOULD also be sent via that tunnel. This allows
any existing "Fast Reroute" schemes to be applied to multicast
traffic as well as to unicast traffic.

Some examples of these forwarding procedures can be found in
Section 6.6.

The rules given in this section can be represented by the following
pseudocode:

```
void ForwardBitMaskPacket (Packet)
{
    SI=GetPacketSI(Packet);
    Offset=SI*BitStringLength;
    for (Index = GetFirstBitPosition(Packet->BitString); Index ;
         Index = GetNextBitPosition(Packet->BitString, Index)) {
        F-BM = BIFT[Index+Offset]->F-BM;
        if (!F-BM) continue;
        BFR-NBR = BIFT[Index+Offset]->BFR-NBR;
        PacketCopy = Copy(Packet);
        PacketCopy->BitString &= F-BM;
        PacketSend(PacketCopy, BFR-NBR);
        Packet->BitString &= ~F-BM;
    }
}
```

Figure 4: Pseudocode

This pseudocode assumes that at a given BFER, the BFR-NBR entry
corresponding to the BFER's own BFR-id will be the BFER's own
BFR-prefix. It also assumes that the corresponding F-BM has only one
bit set, the bit representing the BFER itself. In this case, the
"PacketSend" function sends the packet to the multicast flow overlay.

6.6. Examples of BIER Forwarding

In this section, we give two examples of BIER forwarding, based on
the topology in Figure 1. In these examples, all packets have been
assigned to the default sub-domain, all packets have SI=0, and the
BitStringLength is 4. Figure 5 shows the BIFT entries for SI=0 only.
For compactness, we show the first column of the BIFT, the BFR-id,
only as an integer.

```
        BFR-A BIFT                BFR-B BIFT                BFR-C BIFT
   -------------------       -------------------       -------------------
   | Id | F-BM | NBR |       | Id | F-BM | NBR |       | Id | F-BM | NBR |
   =================       =================       =================
   |  1 | 0111 |  B  |       |  1 | 0011 |  C  |       |  1 | 0001 |  D  |
   -------------------       -------------------       -------------------
   |  2 | 0111 |  B  |       |  2 | 0011 |  C  |       |  2 | 0010 |  F  |
   -------------------       -------------------       -------------------
   |  3 | 0111 |  B  |       |  3 | 0100 |  E  |       |  3 | 1100 |  B  |
   -------------------       -------------------       -------------------
   |  4 | 1000 |  A  |       | 4 |  1000 |  A  |       |  4 | 1100 |  B  |
   -------------------       -------------------       -------------------
```

                   Figure 5: BIFTs for Forwarding Examples

6.6.1. Example 1

   BFR-D, BFR-E and BFR-F are BFER's. BFR-A is the BFIR. Suppose that
   BFIR-A has learned from the multicast flow overlay that BFER-D is
   interested in a given multicast flow. If BFIR-A receives a packet of
   that flow from outside the BIER domain, BFIR-A applies the BIER
   encapsulation to the packet. The encapsulation must be such that the
   SI is zero. The encapsulation also includes a BitString, with just
   bit 1 set and with all other bits clear (i.e., 0001). This indicates
   that BFER-D is the only BFER that needs to receive the packet. Then
   BFIR-A follows the procedures of Section 6.5:

   o Since the packet's BitString is 0001, BFIR-A finds that the first
      bit in the string is bit 1. Looking at entry 1 in its BIFT, BFR-A
      determines that the bit mask F-BM is 0111 and the BFR-NBR is
      BFR-B.

   o BFR-A then makes a copy of the packet, and applies F-BM to the
      copy: Copy->BitString &= 0111. The copy's Bitstring is now 0001
      (0001 & 0111).

   o The copy is now sent to BFR-B.

   o BFR-A then updates the packet's BitString by applying the inverse
      of the F-BM: Packet->Bitstring &= ~F-BM. As a result, the
      packet's BitString is now 0000 (0001 & 1000).

   o As the packet's BitString is now zero, the forwarding procedure is
      complete.

   When BFR-B receives the multicast packet from BFR-A, it follows the
   same procedure. The result is that a copy of the packet, with a
   BitString of 0001, is sent to BFR-C. BFR-C applies the same

procedures, and as a result sends a copy of the packet, with a
BitString of 0001, to BFR-D.

At BFER-D, the BIFT entry (not pictured) for BFR-id 1 will specify an
F-BM of 0000 and a BFR-NBR of BFR-D itself. This will cause a copy
of the packet to be delivered to the multicast flow overlay at BFR-D.
The packet's BitString will be set to 0000, and the packet will not
be forwarded any further.

6.6.2. Example 2

This example is similar to Example 1, except that BFIR-A has learned
from the multicast flow overlay that both BFER-D and BFER-E are
interested in a given multicast flow. If BFIR-A receives a packet of
that flow from outside the BIER domain, BFIR-A applies the BIER
encapsulation to the packet. The encapsulation must be such that the
SI is zero. The encapsulation also includes a BitString with two
bits set: bit 1 is set (as in example 1) to indicate that BFR-D is a
BFER for this packet, and bit 3 is set to indicate that BFR-E is a
BFER for this packet. I.e., the BitString (assuming again a
BitStringLength of 4) is 0101. To forward the packet, BFIR-A follows
the procedures of Section 6.5:

o Since the packet's BitString is 0101, BFIR-A finds that the first
   bit in the string is bit 1. Looking at entry 1 in its BIFT, BFR-A
   determines that the bit mask F-BM is 0111 and the BFR-NBR is
   BFR-B.

o BFR-A then makes a copy of the packet, and applies the F-BM to the
   copy: Copy->BitString &= 0111. The copy's Bitstring is now 0101
   (0101 & 0111).

o The copy is now sent to BFR-B.

o BFR-A then updates the packet's BitString by applying the inverse
   of the F-BM: Packet->Bitstring &= ~F-BM. As a result, the
   packet's BitString is now 0000 (0101 & 1000).

o As the packet's BitString is now zero, the forwarding procedure is
   complete.

When BFR-B receives the multicast packet from BFR-A, it follows the
procedure of Section 6.5, as follows:

o Since the packet's BitString is 0101, BFR-B finds that the first
   bit in the string is bit 1. Looking at entry 1 in its BIFT, BFR-B
   determines that the bit mask F-BM is 0011 and the BFR-NBR is
   BFR-C.

o BFR-B then makes a copy of the packet, and applies the F-BM to the
   copy: Copy->BitString &= 0011. The copy's Bitstring is now 0001
   (0101 & 0011).

o The copy is now sent to BFR-C.

o BFR-B then updates the packet's BitString by applying the inverse
   of the F-BM: Packet->Bitstring &= F-BM. As a result, the
   packet's BitString is now 0100 (0101 & 1100).

o Now BFR-B finds the next bit in the packet's (modified) BitString.
   This is bit 3. Looking at entry 3 in its BIFT, BFR-B determines
   that the F-BM is 0100 and the BFR-NBR is BFR-E.

o BFR-B then makes a copy of the packet, and applies the F-BM to the
   copy: Copy->BitString &= 0100. The copy's Bitstring is now 0100
   (0100 & 0100).

o The copy is now sent to BFR-E.

o BFR-B then updates the packet's BitString by applying the inverse
   of the F-BM: Packet->Bitstring &= ~F-BM. As a result, the
   packet's BitString is now 0000 (0100 & 1011).

o As the packet's BitString is now zero, the forwarding procedure is
   complete.

Thus BFR-B forwards two copies of the packet. One copy of the
packet, with BitString 0001, has now been sent from BFR-B to BFR-C.
Following the same procedures, BFR-C will forward the packet to
BFER-D.

At BFER-D, the BIFT entry (not pictured) for BFR-id 1 will specify an
F-BM of 0000 and a BFR-NBR of BFR-D itself. This will cause a copy
of the packet to be delivered to the multicast flow overlay at BFR-D.
The packet's BitString will be set to 0000, and the packet will not
be forwarded any further.

The other copy of the packet has been sent from BFR-B to BFER-E, with
BitString 0100.

At BFER-E, the BIFT entry (not pictured) for BFR-id 3 will specify an
F-BM of 0000 and a BFR-NBR of BFR-E itself. This will cause a copy
of the packet to be delivered to the multicast flow overlay at BFR-E.
The packet's BitString will be set to 0000, and the packet will not
be forwarded any further.

6.7. Equal Cost Multi-path Forwarding

   In many networks, the routing underlay will provide multiple equal
   cost paths from a given BFR to a given BFER. When forwarding
   multicast packets through the network, it can be beneficial to take
   advantage of this by load balancing among those paths. This feature
   is known as "equal cost multiple path forwarding", or "ECMP".

   BIER supports ECMP, but the procedures of Section 6.5 must be
   modified slightly. Two ECMP procedures are defined. In the first
   (described in Section 6.7.1), the choice among equal-cost paths taken
   by a given packet from a given BFR to a given BFER depends on (a) the
   packet's entropy, and (b) the other BFERs to which that packet is
   destined. In the second (described in Section 6.7.2), the choice
   depends only upon the packet's entropy.

   There are tradeoffs between the two forwarding procedures described
   here. In the procedure of Section 6.7.1, the number of packet
   replications is minimized. The procedure in Section 6.7.1 also uses
   less memory in the BFR. In the procedure of Section 6.7.2, the path
   traveled by a given packet from a given BFR to a given BFER is
   independent of the other BFERs to which the packet is destined.
   While the procedures of Section 6.7.2 may cause more replications,
   they provide a more predictable behavior.

   The two procedures described here operate on identical packet formats
   and will interoperate correctly. However, if deterministic behavior
   is desired, then all BFRs would need to use the procedure from
   Section 6.7.2.

6.7.1. Non-deterministic ECMP

   Figure 6 shows the operation of non-deterministic ECMP in BIER.

```
      BFR-A BIFT              BFR-B BIFT              BFR-C BIFT
  ------------------      ------------------      ------------------
 | Id | F-BM | NBR |     | Id | F-BM | NBR |     | Id | F-BM | NBR |
 ==================      ==================      ==================
 | 1  | 0111 |  B  |     | 1  | 0011 |  C  |     | 1  | 0001 |  D  |
  ------------------      ------------------      ------------------
 | 2  | 0111 |  B  |     | 2  | 0011 |  C  |     | 2  | 0010 |  F  |
  ------------------     |    | 0110 |  E  |      ------------------
 | 3  | 0111 |  B  |      ------------------     | 3  | 1100 |  B  |
  ------------------     | 3  | 0110 |  E  |      ------------------
 | 4  | 1000 |  A  |      -----------------|     | 4  | 1100 |  B  |
  ------------------     | 4  | 1000 |  A  |      ------------------
                          ------------------

   ( A ) ----------- (  B  ) ----------- ( C ) ----------- ( D )
  4 (0:1000)                \                  \             1 (0:0001)
                             \                  \
                            ( E ) ----------- ( F )
                           3 (0:0100)        2 (0:0010)
```
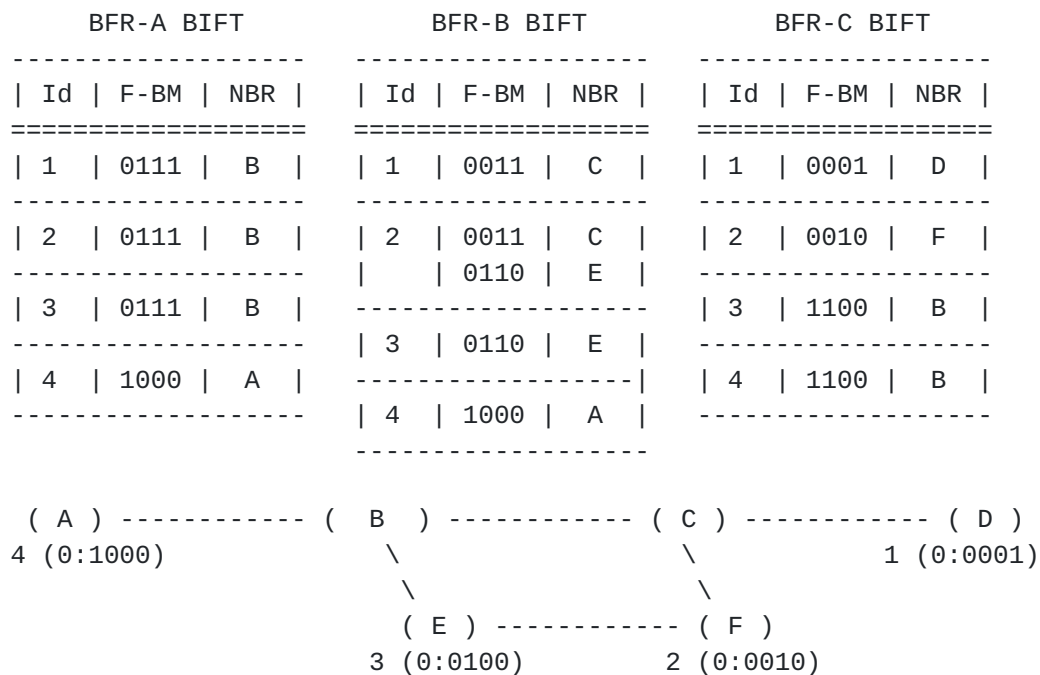
                    Figure 6: Example of ECMP

   In this example, BFR-B has two equal cost paths to reach BFER-F, one
   via BFR-C and one via BFR-E. Since the BFR-id of BFER-F is 2, this
   is reflected in entry 2 of BFR-B's BIFT. Entry 2 shows that BFR-B
   has a choice of two BFR-NBRs for BFER-B, and that a different F-BM is
   associated with each choice. When BFR-B looks up entry 2 in the
   BIFT, it can choose either BFR-NBR. However, when following the
   procedures of Section 6.5, it MUST use the F-BM corresponding to the
   BFR-NBR that it chooses.

   How the choice is made is an implementation matter. However, the
   usual rules for ECMP apply: packets of a given flow SHOULD NOT be
   split among two paths, and any "entropy" field in the packet's
   encapsulation SHOULD be respected.

   Note however that by the rules of Section 6.5, any packet destined
   for both BFER-D and BFER-F will be sent via BFR-C.

6.7.2. Deterministic ECMP

   With the procedures of Section 6.7.1, where ECMP paths exist, the
   path a packet takes to reach any particular BFER depends not only on
   routing and on the packet's entropy, but also on the set of other
   BFERs to which the packet is destined.

   For example consider the following scenario in the network of
   Figure 6.

o There is a sequence of packets being transmitted by BFR-A, some of
   which are destined for both D and F, and some of which are
   destined only for F.

o All the packets in this sequence have the same entropy value, call
   it "Q".

o At BFR-B, when a packet with entropy value Q is forwarded via
   entry 2 in the BIFT, the packet is sent to E.

Using the forwarding procedure of Section 6.7.1, packets of this
sequence that are destined for both D and F are forwarded according
to entry 1 in the BIFT, and thus will reach F via the path A-B-C-F.
However, packets of this sequence that are destined only for F are
forwarded according to entry 2 in the BIFT, and thus will reach F via
the path A-B-E-F.

That procedure minimizes the number of packets transmitted by BFR B.
However, consider the following scenario:

o Beginning at time t0, the multicast flow in question needs to be
   received ONLY by BFER-F;

o Beginning at a later time, t1, the flow needs to be received by
   both BFER-D and BFER-F.

o Beginning at a later time, t2, the no longer needs to be received
   by D, but still needs to be received by F.

Then from t0 until t1, the flow will travel to F via the path
A-B-E-F. From t1 until t2, the flow will travel to F via the path
A-B-C-F. And from t2, the flow will again travel to F via the path
A-B-E-F.

The problem is that if D repeatedly joins and leaves the flow, the
flow's path from B to F will keep switching. This could cause F to
receive packets out of order. It also makes troubleshooting
difficult. For example, if there is some problem on the E-F link,
receivers at F will get good service when the flow is also going to D
(avoiding the E-F link), but bad service when the flow is not going
to D. Since it is hard to know which path is being used at any given
time, this may be hard to troubleshoot. Also, it is very difficult
to perform a traceroute that is known to follow the path taken by the
flow at any given time.

The source of this difficulty is that, in the procedures of
Section 6.7.1, the path taken by a particular flow to a particular
BFER depends upon whether there are lower numbered BFERs that are

also receiving the flow. Thus the choice among the ECMP paths is
fundamentally non-deterministic.

Deterministic forwarding can be achieved by using multiple BIFTs,
such that each row in a BIFT has only one path to each destination,
but the multiple ECMP paths to any particular destination are spread
across the multiple tables. When a BIER-encapsulated packet arrives
to be forwarded, the BFR uses a hash of the BIER Entropy field to
determine which BIFT to use, and then the normal BIER forwarding
algorithm (as described in Sections 6.5 and 6.6) is used with the
selected BIFT.

As an example, suppose there are two paths to destination X (call
them X1 and X2), and four paths to destination Y (call them Y1, Y2,
Y3, and Y4). If there are, say, four BIFTs, one BIFT would have
paths X1 and Y1, one would have X1 and Y2, one would have X2 and Y3,
and one would have X2 and Y4. If traffic to X is split evenly among
these four BIFTs, the traffic will be split evenly between the two
paths to X; if traffic to Y is split evenly among these four BIFTs,
the traffic will be split evenly between the four paths to Y.

Note that if there are three paths to one destination and four paths
to another, 12 BIFTs would be required in order to get even splitting
of the load to each of those two destinations. Of course, each BIFT
uses some memory, and one might be willing to have less optimal
splitting in order to have fewer BIFTs. How that tradeoff is made is
an implementation or deployment decision.

6.8. Prevention of Loops and Duplicates

The BitString in a BIER-encapsulated packet specifies the set of
BFERs to which that packet is to be forwarded. When a BIER-
encapsulated packet is replicated, no two copies of the packet will
ever have a BFER in common. If one of the packet's BFERs forwards
the packet further, that will first clear the bit that identifies
itself. As a result, duplicate delivery of packets is not possible
with BIER.

As long as the routing underlay provides a loop free path between
each pair of BFRs, BIER-encapsulated packets will not loop. Since
the BIER layer does not create any paths of its own, there is no need
for any BIER-specific loop prevention techniques beyond the
forwarding procedures specified in Section 6.5.

If, at some time, the routing underlay is not providing a loop free
path between BFIR-A and BFER-B, then BIER encapsulated packets may
loop while traveling from BFIR-A to BFER-B. However, such loops will
never result in delivery of duplicate packets to BFER-B.

These properties of BIER eliminate the need for the "reverse path
forwarding" (RPF) check that is used in conventional IP multicast
forwarding.

6.9. When Some Nodes do not Support BIER

The procedures of section Section 6.2 presuppose that, within a given
BIER domain, all the nodes adjacent to a given BFR in a given routing
underlay are also BFRs. However, it is possible to use BIER even
when this is not the case, as long as the ingress and egress nodes
are BFRs. In this section, we assume that the routing underlay is an
SPF-based IGP that computes a shortest path tree from each node to
all other nodes in the domain.

At a given BFR, say BFR B, start with a copy of the IGP-computed
shortest path tree from BFR B to each router in the domain. (This
tree is computed by the SPF algorithm of the IGP.) Let's call this
copy the "BIER-SPF tree rooted at BFR B." BFR B then modifies this
BIER-SPF tree as follows.

1. BFR B looks in turn at each of B's child nodes on the BIER-SPF
    tree.

2. If one of the child nodes does not support BIER, BFR B removes
    that node from the tree. The child nodes of the node that has
    just been removed are then re-parented on the tree, so that BFR B
    now becomes their parent.

3. BFR B then continues to look at each of its child nodes,
    including any nodes that have been re-parented to B as a result
    of the previous step.

When all of the child nodes (the original child nodes plus any new
ones) have been examined, B's children on the BIER-SPF tree will all
be BFRs.

When the BIFT is constructed, B's child nodes on the BIER-SPF tree
are considered to be the BFR-NBRs. The F-BMs and outgoing BIER-MPLS
labels must be computed appropriately, based on the BFR-NBRs.

B may now have BFR-NBRs that are not "directly connected" to B via
layer 2. To send a packet to one of these BFR-NBRs, B will have to
send the packet through a unicast tunnel. This may be as simple as
finding the IGP unicast next hop to the child node, and pushing on
(above the BIER-MPLS label advertised by the child) the MPLS label
that the IGP next hop has bound to an address of the child node. (If
for some reason the unicast tunnel cannot be an MPLS tunnel, any

other kind of tunnel can be used, as long as it is possible to
encapsulate MPLS within that kind of tunnel.)

Of course, the above is not meant as an implementation technique,
just as a functional description.

While the above description assumes that the routing underlay
provides an SPF tree, it may also be applicable to other types of
routing underlay.

Note that the technique above can also be used to provide "node
protection" (i.e., to provide fast reroute around nodes that are
believed to have failed). If BFR B has a failed BFR-NBR, B can
remove the failed BFR-NBR from the BIER-SPF tree, and can then re-
parent the child BFR-NBRs of the failed BFR-NBR so that they appear
to be B's own child nodes on the tree (i.e., so that they appear to
be B's BFR-NBRs). Then the usual BIER forwarding procedures apply.
However, getting the packet from B to the child nodes of the failed
BFR-NBR is a bit more complicated, as it may require using a unicast
bypass tunnel to get around the failed node.

A simpler variant of step 2 above would be the following:

   If one of the child nodes does not support BIER, BFR B removes
   that node from the tree. All BFERs that are reached through that
   child node are then re-parented on the tree, so that BFR B now
   becomes their parent.

This variant is simpler because the set of BFERs that are reached
through a particular child node of B can be determined from the F-BM
in the BIFT. However, if this variant is used, the results are less
optimal, because packets will be unicast directly from B to the BFERs
that are reachable through the non-BIER child node.

When using a unicast MPLS tunnel to get a packet to a BFR-NBR, it may
be advantageous to (a) set the TTL of the MPLS label entry
representing the "tunnel" to a large value, rather than copying the
TTL value from the BIER-MPLS label, and (b) when the tunnel labels
are popped off, to avoid copying the TTL from the tunnel labels to
the BIER-MPLS label. That way, the TTL of the BIER-MPLS label would
only control the number of "BFR hops" that the packet may traverse.

The material in this section presupposes that a given node is either
a BFR or not, and that a BFR supports BIER on all its interfaces. It
is however possible that a router will have some line cards that
support BIER and some that do not. In such a case, one can think of
the router as a "partial-BFR", that supports BIER only on some of its
interfaces. If it is desired to deploy such partial-BFRs, one can

use the multi-topology features of the IGP to set up a BIER-specific
topology. This topology would exclude all the non-BIER-capable
interfaces that attach to BFRs. BIER would then have to be run in a
sub-domain that is bound to this topology. If unicast tunnels are
used to bypass non-BFRs, either the tunnels have to be restricted to
this topology, or the tunnel endpoints have to be BFRs that do not
have any non-BIER-capable interfaces.

6.10. Use of Different BitStringLengths within a Domain

It is possible for different BFRs within a BIER domain to be using
different Imposition and/or Disposition BitStringLengths. As stated
in Section 3:

   "if a particular BFIR is provisioned to use a particular
   Imposition BitStringLength and a particular Imposition sub-domain
   when imposing the encapsulation on a given set of packets, all
   other BFRs with BFR-ids in that sub-domain SHOULD be provisioned
   to process received BIER packets with that BitStringLength (i.e.,
   all other BFRs with BFR-ids in that sub-domain SHOULD be
   provisioned with that BitStringLength as a Disposition
   BitStringLength for that sub-domain)."

Note that mis-provisioning can result in "black holes". If a BFIR
creates a BIER packet with a particular BitStringLength, and if that
packet needs to travel through a BFR that cannot process received
BIER packets with that BitStringLength, then it may be impossible to
forward the packet to all of the BFERs identified in its BIER header.
Section 6.10.1 defines a procedure, the "BitStringLength
Compatibility Check", that can be used to detect the possibility of
such black holes.

However, failure of the BitStringLength Compatibility Check does not
necessarily result in the creation of black holes; Section 6.10.2
specifies OPTIONAL procedures that allow BIER forwarding to proceed
without black holes, even if the BitStringLength Compatibility Check
fails.

If the procedures of Section 6.10.2 are not deployed, but the
BitStringLength Compatibility Check fails at some BFIR, the BFIR has
two choices:

o Create BIER packets with the provisioned Imposition
   BitStringLength, even though the packets may not be able to reach
   all the BFERs identified in their BitStrings

o Use an Imposition BitStringLength that passes the Compatibility
   Check (assuming that there is one), even if this is not the
   provisioned Imposition BitStringLength.

Section 6.10.1 discusses the implications of making one or the other
of these choices.

There will be times when an operator wishes to change the
BitStringLengths used in a particular BIER domain. Section 6.10.3
specifies a simple procedure that can be used to transition a BIER
domain from one BitStringLength to another.

6.10.1. BitStringLength Compatibility Check

When a BFIR needs to encapsulate a packet, the BFIR first assigns the
packet to a sub-domain. Then the BFIR chooses an Imposition
BitStringLength L for the packet. The choice of Imposition
BitStringLength is by provisioning. However, the BFIR should also
perform the BitStringLength Compatibility Check defined below.

The combination of Sub-Domain S and Imposition BitStringLength L
passes the BitStringLength Compatibility Check if and only if the
following condition holds:

   Every BFR that has advertised its membership in sub-domain S has
   also advertised that it is using Disposition BitStringLength L
   (and possibly other BitStringLengths as well) in that Sub-Domain.
   (If the MPLS encapsulation [MPLS_BIER_ENCAPS] is being used, this
   means that every BFR that is advertising a label for Sub-Domain S
   is advertising a label for the combination of Sub-Domain S and
   Disposition BitStringLength L.)

If a BFIR has been provisioned to use a particular Imposition
BitStringLength and a particular sub-domain for some set of packets,
and if that combination of Imposition BitStringLength and sub-domain
does not pass the BitStringLength Compatibility Check, the BFIR
SHOULD log this fact as an error. It then has the following choice
about what to do with the packets:

1. The BFIR MAY use the provisioned Imposition BitStringLength
    anyway. If the procedure Paragraph 2 or Paragraph 3 of
    Section 6.10.2 are deployed, this will not cause black holes, and
    may actually be the optimal result. It should be understood
    though that the BFIR cannot determine by signaling whether those
    procedures have been deployed.

2. If the BFIR is capable of using an Imposition BitStringlength
    that does pass the BitStringLength Compatibility Check for the

particular sub-domain, the BFIR MAY use that Imposition
BitStringLength instead.

Which of these two choices to make is itself determined by
provisioning.

Note that discarding the packets is not one of the allowable choices.
Suppose, for example, that all the BFIRs are provisioned to use
Imposition BitStringLength L for a particular sub-domain S, but one
BFR has not been provisioned to use Disposition BitStringLength L for
sub-domain S. This will cause the BitStringLength Compatibility
Check to fail. If the BFIR sends packets with BitStringLength L and
sub-domain S, the mis-provisioned BFR will not be able to forward
those packets, and thus the packets may only be able to reach a
subset of the BFERs to which they are destined. However, this is
still better than having the BFIRs drop the packets; if the BFIRs
discard the packets, the packets won't reach any of the BFERs to
which they are destined at all.

If the procedures of Section 6.10.2 have not been deployed, choice 2
might seem like a better option. However, there might not be any
Imposition BitStringLength that a given BFIR can use that also passes
the BitStringLength Compatibility Check. If it is desired to use
choice 2 in a particular deployment, then there should be a "Fallback
Disposition BitStringLength", call it F, such that:

o Every BFR advertises that it uses BitStringLength F as a
   Disposition BitStringLength for every sub-domain, and

o If a BFIR is provisioned to use Imposition BitStringLength X and
   Imposition sub-domain S for a certain class of packets, but the
   BitStringLength Compatibility check fails for the combination of
   BitStringLength X and sub-domain S, then the BFIR will fall back
   to using BitStringLength F as the Imposition BitStringLength
   whenever the Imposition sub-domain is S.

This fallback procedure will work best if the value of F is
established by the architecture, rather than by provisioning.

6.10.2. Handling BitStringLength Mismatches

Suppose a packet has been BIER-encapsulated with a BitStringLength
value of X, and that the packet has arrived at BFR-A. Now suppose
that according to the routing underlay, the next hop is BFR-B, but
BFR-B is not using X as one of its Disposition BitStringLengths.
What should BFR-A do with the packet? BFR-A has three options. It
MUST do one of the three, but the choice of which procedure to follow
is a local matter. The three options are:

1. BFR-A MAY discard the packet.

2. BFR-A MAY re-encapsulate the packet, using a BIER header whose
    BitStringLength value is supported by BFR-B.

    Note that if BFR-B only uses Disposition BitStringLength values
    that are smaller than the BitStringLength value of the packet,
    this may require creating additional copies of the packet.
    Whether additional copies actually have to be created depends
    upon the bits that are actually set in the original packet's
    BitString.

3. BFR-A MAY treat BFR-B as if BFR-B did not support BIER at all,
    and apply the rules of Section 6.9.

Note that there is no signaling that enables a BFR to advertise which
of the three options it will use.

Option 2 can be useful if there is a region of the BIER domain where
the BFRs are capable of using a long BitStringLength, and a region
where the BFRs are only capable of using a shorter BitStringLength.

6.10.3. Transitioning from One BitStringLength to Another

Suppose one wants to migrate the BitStringLength used in a particular
BIER domain from one value (X) to another value (Y). The following
migration procedure can be used. This procedure allows the BFRs to
be reprovisioned one at a time, and does not require a "flag day".

First, upgrade all the BFRs in the domain so that they use both value
X and value Y as their Disposition BitStringLengths. Once this is
done, reprovision the BFIRs so that they use BitStringLength value Y
as the Imposition BitStringLength. Once that is done, one may
optionally reprovision all the BFRs so that they no longer use
Dispostion BitStringLength X.

7. IANA Considerations

This document contains no actions for IANA.

8. Security Considerations

When BIER is paired with a particular multicast flow overlay, it
inherits the security considerations of that layer. Similarly, when
BIER is paired with a particular routing underlay, it inherits the
security considerations of that layer.

If the BIER encapsulation of a particular packet specifies an SI or a
BitString other than the one intended by the BFIR, the packet is
likely to be misdelivered. If the BIER encapsulation of a packet is
modified (through error or malfeasance) in a way other than that
specified in this document, the packet may be misdelivered.

If the procedures used for advertising BFR-ids and BFR-prefixes are
not secure, an attack on those procedures may result in incorrect
delivery of BIER-encapsulated packets.

Every BFR must be provisioned to know which of its interfaces lead to
a BIER domain and which do not. If two interfaces lead to different
BIER domains, the BFR must be provisioned to know that those two
interfaces lead to different BIER domains. If the provisioning is
not correct, BIER-encapsulated packets from one BIER domain may
"leak" into another; this is likely to result in misdelivery of
packets.

9. Acknowledgements

The authors wish to thank Rajiv Asati, John Bettink, Ross Callon (who
contributed much of the text on deterministic ECMP), Nagendra Kumar,
Christian Martin, Neale Ranns, Greg Shepherd, Albert Tian, Ramji
Vaithianathan, Xiaohu Xu and Jeffrey Zhang for their ideas and
contributions to this work.

10. Contributor Addresses

Below is a list of other contributing authors in alphabetical order:

Gregory Cauchie
Bouygues Telecom

Email: gcauchie@bouyguestelecom.fr

Mach (Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Arkadiy Gulko
Thomson Reuters
195 Broadway
New York NY 10007
United States

Email: arkadiy.gulko@thomsonreuters.com

   Wim Henderickx
   Alcatel-Lucent
   Copernicuslaan 50
   Antwerp 2018
   Belgium

   Email: wim.henderickx@alcatel-lucent.com

   Martin Horneffer
   Deutsche Telekom
   Hammer Str. 216-226
   Muenster 48153
   Germany

   Email: Martin.Horneffer@telekom.de

   Uwe Joorde
   Deutsche Telekom
   Hammer Str. 216-226
   Muenster D-48153
   Germany

   Email: Uwe.Joorde@telekom.de

   Luay Jalil
   Verizon
   1201 E Arapaho Rd.
   Richardson, TX 75081
   United States

   Email: luay.jalil@verizon.com

   Jeff Tantsura
   Ericsson
   300 Holger Way
   San Jose, CA 95134
   United States

   Email: jeff.tantsura@ericsson.com

11. References

11.1. Normative References

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

11.2. **Informative References**

[Boivie_Feldman]
          Boivie, R. and N. Feldman, "Small Group Multicast",
          (expired) draft-boivie-sgm-02.txt, February 2001.

[ISIS_BIER_EXTENSIONS]
          Ginsberg, L., Przygienda, T., Aldrin, S., and J. Zhang,
          "BIER Support via ISIS", internet-draft draft-ietf-bier-
          isis-ranges-00.txt, April 2015.

[MPLS_BIER_ENCAPS]
          Wijnands, IJ., Rosen, E., Dolganow, A., Tantsura, J., and
          S. Aldrin, "Encapsulation for Bit Index Explicit
          Replication in MPLS Networks", internet-draft draft-ietf-
          bier-mpls-encapsulation-00.txt, April 2015.

[OSPF_BIER_EXTENSIONS]
          Psenak, P., Kumar, N., Wijnands, IJ., Dolganow, A.,
          Przygienda, T., and J. Zhang, "OSPF Extensions for Bit
          Index Explicit Replication", internet-draft draft-ietf-
          ospf-bier-extensions-00.txt, April 2015.

[RFC6513] Rosen, E. and R. Aggarwal, "Multicast in MPLS/BGP IP
          VPNs", RFC 6513, February 2012.

[RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP
          Encodings and Procedures for Multicast in MPLS/BGP IP
          VPNs", RFC 6514, February 2012.

Authors' Addresses

   IJsbrand Wijnands (editor)
   Cisco Systems, Inc.
   De Kleetlaan 6a
   Diegem 1831
   Belgium

   Email: ice@cisco.com

   Eric C. Rosen (editor)
   Juniper Networks, Inc.
   10 Technology Park Drive
   Westford, Massachusetts 01886
   United States

   Email: erosen@juniper.net

   Andrew Dolganow
   Alcatel-Lucent
   600 March Rd.
   Ottawa, Ontario K2K 2E6
   Canada

   Email: andrew.dolganow@alcatel-lucent.com

   Tony Przygienda
   Ericsson
   300 Holger Way
   San Jose, California 95134
   United States

   Email: antoni.przygienda@ericsson.com

   Sam K Aldrin
   Google, Inc.
   1600 Amphitheatre Parkway
   Mountain View, California
   United States

   Email: aldrin.ietf@gmail.com