

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 6, 2018

T. Eckert
Huawei
Mar 5, 2018

Framework for Traffic Engineering with BIER-TE forwarding (Bit Index
Explicit Replication with Traffic Engineering)
draft-eckert-teas-bier-te-framework-00

Abstract

BIER-TE is an application-state free, (loose) source routed multicast forwarding method where every hop and destination is identified via bits in a bitstring of the data packets. It is described in [I-D.ietf-bier-te-arch]. BIER-TE is a variant of [RFC8279] in support of such explicit path engineering.

This document describes the traffic engineering control framework for use with the BIER-TE forwarding plane: How to enable the ability to calculate paths and integrate this forwarding plane into an overall TE solution.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction and Overview	2
2. BIER-TE Topology management	6
2.1. Operational model	6
2.2. BIER-TE topology model	7
2.3. Consistency checking	10
2.4. Auto-configuration	11
3. Flow Management	12
3.1. Operational / Architectural Models	12
3.1.1. Overprovisioning	13
3.1.2. PCEC	13
3.1.2.1. per-flow QoS - policer/shaper/EF	14
3.1.2.2. DiffServ QoS	15
3.2. BIER-TE flow model	15
4. Security Considerations	17
5. IANA Considerations	17
6. Acknowledgements	17
7. Change log [RFC Editor: Please remove]	17
8. References	18
Author's Address	19

1. Introduction and Overview

This document proposes a framework and abstract data model for the control plane of BIER-TE as defined in [I-D.ietf-bier-te-arch] (BIER-TE-ARCH). That document primarily defines the forwarding plane and provides some example scenarios how to use it.

BIER-TE is a forwarding plane derived from BIER ([RFC8279]) in which the destinations of packets are bits in a bitstring. Every bit indicates a destination (BFER - BIER Forwarding Exit Router) and an IGP is used to flood those "bit addresses" so hops along the path from sender (BFIR - BIER Forwarding Ingress Router) through intermediate nodes (BFR) can calculate the shortest path for each destination (bit) and simply copy the received packet to every interface to one or more bits set in the packet.

In BIER-TE, shortest path calculation is replaced by bits of the bitstring indicating intermediate hops and pre-populated forwarding tables (BIFT - Bit Index Forwarding Tables) on every BFR indicating

those bits. In the simplest case, every interface on a BFR has a unique bit assigned to it, and the BIFT of only that BFR will have in its BIFT for this bit an adjacency entry indicating that interface. This ultimately allows to indicate any sub-graph of the network topology as a bitstring and hop-by-hop perform the necessary forwarding/replication for a packet with such a bitstring. More complex semantics of bits are used to help saving bits. A typical bitstring size supportable is 256 bits, the original BIER specification allows up to 1024 bits. BIER-TE may be specifically interesting for typically smaller topologies such as often encountered in DetNet scenarios, or else through intelligent allocating and saving of bits for larger topologies, some of which is exemplified in BIER-TE-ARCH.

One can compare BIER-TE in function to Segment Routing in so far that it attempts to be as much as possible a per-packet "source-routed" (for lack of better term) forwarding paradigm without per-application/flow state in the network. Whereas SR primarily supports simple sequential paths indicated as a sequence of SIDs, in BIER-TE, the bitstring indicate a directed and acyclic graphs (DAG) - with replications. BIER-TE can also be combined with SR and then bits in the bitstring are only required for the nodes (BFR) where replication is desired, and the paths between any two such replication nodes could be SIDs or stack of SIDs that are selected by assigning bits to them (routed adjacencies in the BIER-TE terminology).

In BIER-TE-ARCH, the control plane is not considered. In its place, a theoretical BIER-TE Controller Host uses unspecified signaling to control the setup of the BIER-TE forwarding-plane end to end (all bits/adjacencies in all BFR BIFTs) and during the lifecycle of network device install through the determination of paths for specific traffic and changes to the topology. This document expands and refines this simplistic model and intends to serve as the framework for follow-up protocol and data model specification work.

The core forwarding documents relevant to this document are as follows:

- o [RFC8279] (BIER-ARCH): as summarized above.
- o [RFC8296] (BIER-ENCAP): The encapsulation for BIER packets using MPLS or non-MPLS networks underneath.
- o [I-D.ietf-bier-te-arch] (BIER-TE-ARCH): as summarized above.
- o [I-D.thubert-bier-replication-elimination] (BIER-EF-OAM): Extends the BIER-TE forwarding from BIER-TE-ARCH to support the Elimination Function (EF) and an OAM function. The Elimination

Function is a term from DetNets resilience architecture: Multiple copies of traffic flows are carried across disjoint path, merged in a BFR running the EF and duplicates are eliminated on that BFR based on recognizing duplicate sequence numbers. Engineered multiple transmission paths are a key reason to leverage BIER-TE.

- o [I-D.huang-bier-te-encapsulation] (BIER-TE-ENCAP): Proposed encapsulation based on an extension of BIER-ENCAP. Identifies whether the packet expects to use a BIER or BIER-TE BIFT. Also adds a control-word in support of (optional) elimination function (EF) and interprets the pre-existing BFIR-ID and entropy fields as a flow-id.
- o [I-D.eckert-bier-te-frr] (BIER-TE-FRR): This document describes protections methods applicable to BIER-TE. 1:1 / end-to-end path protection is referenced in this document in the context of DetNet style PREF path protection. The options not discussed yet (TBD) in this document are link protection tunnels (such as used in RSVP-TE as well) and the novel BIER-TE specific protection method, in which nodes modify the bitstring upon local discovery of a failure.

The relevant routing underlay documents are as follows:

- o [I-D.ietf-bier-isis-extensions] (BIER-ISIS), [I-D.ietf-bier-ospf-bier-extensions] (BIER-OSPF): The BIER-ISIS and BIER-OSPF documents describe extensions to those two IGPs in support of BIER. Effectively, every BFR announces the <SD,SI-range> BIFTs it is configured for, the MT-ID (IGP Multitopology-ID) they are using, and the BFR-ID it has in each SD (none if it does not need to operate as a BFER). For MPLS encapsulation, the base label for every SD is announced as well as the SI-range (one label per <SD,SI> is used).
- o There is currently no document describing IGP extensions for BIER-TE, but the goal is to define those based, using the proposals made in this framework, and as feasible re-using and/or amending those existing BIER IGP extensions.
- o [I-D.ietf-bier-bier-yang] (BIER-YANG): This document describes the YANG data model to provision on every BFR BIER. It also provides OAM functions. There is currently no model expanding this to support BIER-TE. This framework document defines elements that should be included in a BIER-TE YANG model.
- o TBD: incomplete list ?.

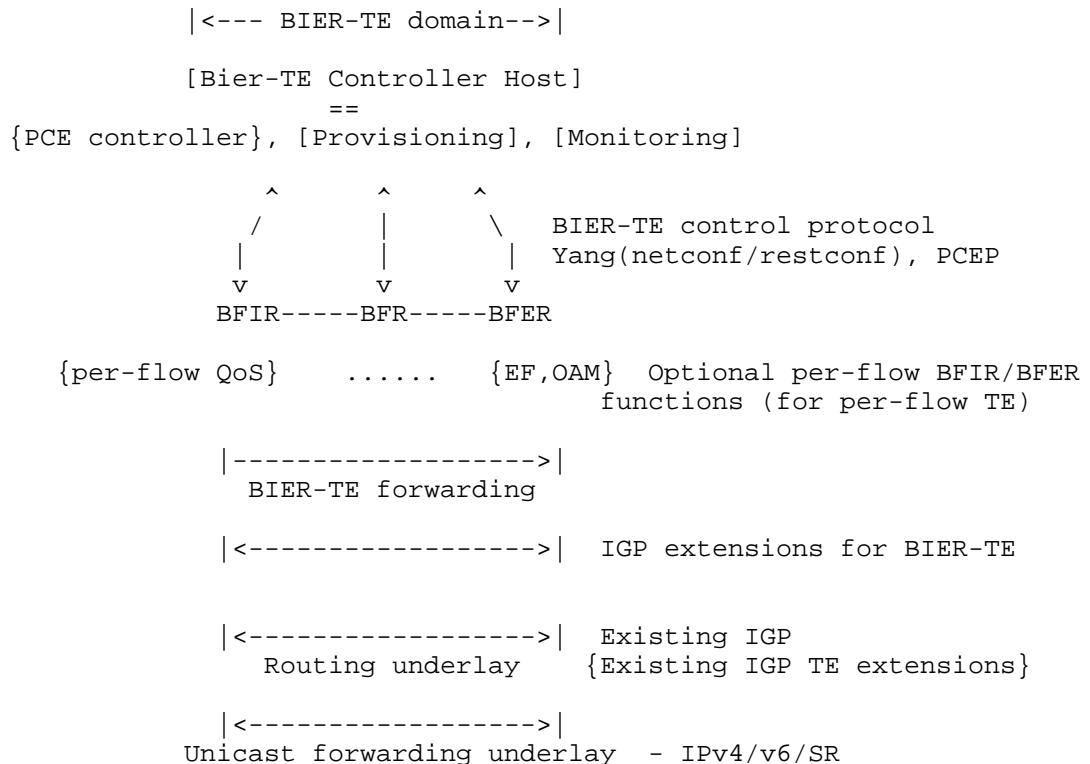


Figure 1: BIER-TE signaling architecture

The above picture is a modified version of Picture 2 from BIER-TE-ARCH reduced by the elements not considered in this document, and refined with those that are intended to be described by this document.

In comparison with BIER-TE-ARCH, Picture 2, this picture and this document do not include considerations for specific multicast flow overlay elements. Instead, it adds description of optional BFIR/BFER elements for per-flow QoS/EF (Elimination Function) and OAM, which are optional parts of an overall BIER-TE traffic engineering architecture. See BIER-EF-OAM for more background.

The routing underlay is refined in this document to consider a unicast forwarding underlay of IPv4/IPv6 and/or unicast SR (Segment Routing) for BIER-TE "forward_routed" adjacencies. It also assumes an existing IGP, such as ISIS or OSPF as the routing underlay. This may include (TBD) extensions already supporting TE aspects (like those IGP extensions done for RSVP-TE).

This framework intends to support a wide range of options to instantiate it:

In one extreme (PCEC only), there is no IGP in the network that BIER-TE depends on, but all BIER-TE operations is managed in an SDN-style fashion from centralized components called "BIER-TE Controller Host" in BIER-TE-ARCH. This central packend can be further subdivided into a Configuration/Provisioning component to install the BIER-TE topology into the network and a PCEC (Pat Computation Engine Controller) and (TBD) monitoring components. After BIER-TE is operational, the PCEC calculates BIER-TE bitstrings for BFIR when they need to send traffic flow to

In the other extreme (IGP only), there is no need for a PCEC or NMS. The initial setup of the BIER-TE topology can be performed manually, using configuration options to support automatic consistency checking and partial auto-configuration to simplify this work. BIER-TE extensions of the IGP are used for consistency checking and autoconfiguration and finally to provide the whole BIER-TE topology to BFIR that can then autonomously calculate BIER-TE bitstrings without the help of a PCEC.

2. BIER-TE Topology management

2.1. Operational model

When a network is installed, BIER-TE is added as a service or later when it is meant to change, BFR need to be (re)provisioned. This involves a planning phase which physical adjacencies (links) should be used in the BIER-TE topology, and which virtual adjacencies (routed adjacencies) should be created and assigned bits. Ultimately this means the definition of the BIER-TE topology.

When the physical topology if the network is smaller than the possible bitstring size (e.g.: 256 bits), then this can be a simple, fully automated process. Likewise, if multiple disjointed services for BIER-TE each require active subsets of the network topology smaller than the network topology, it likewise can be simple to create a different SD (subdomain) BIER-TE topologies for each such service.

When the required network topology for a BIER-TE service exceeds the supportable bitstring size, bit-saving mechanisms can be employed as described in BIER-ARCH. Some of them such as p2p link bits or lan-bits are easily automatically calculated. Creation of virtual adjacencies (routed adjacencies) may likely best be done with operator defined policies applied to a system a system calculating the bits for the BIER-TE topology.

Ultimately, if the set of required destinations plus transit hops exceeds the size of available bitstrings after optimization, multiple BIFT == bitstrings need to be allocated to support this case. These multiple BIFT will likely need to be engineered to minimize duplicate traffic load on the network and minimize bit use. One example shown in BIER-TE-ARCH is to allocate different <SD,SI> BIFT to different areas of a network, therefore having to create one BIER-TE packet copy per required destination region, but in result having only one packet copy in each of those regions.

Provisioning / initial setup can be done manually in simpler networks or through a provisioning system. A PCEP may equally perform this function. If a PCEP is not used to perform this function, but a PCEP is used later for Flow Management, then the PCEP does of course need to also learn the BIER-TE topologies created by the provisioning system.

Unless a PCEC is used for provisioning/initial setup, YANG is likely the preferred model to install the BIER-TE topology information into the BFR. If a PCEC is used, YANG or PCEC seem to be valid choices.

When the network topology expands, bit assignments for the new parts of the topology need to be made. If expansion was not factored into the initial bit assignment plans, this can lead to the need to reassign bits for existing parts of the topology. Support for such processes could be simplified through additional topology information, for example to enable seamless switching of traffic flows from bits in one SD over to bits in another SD. This is currently not considered in this document.

2.2. BIER-TE topology model

```

<BFR> BIFT information:
  Instance: "configured", "operational",
            "learned-configured", "learned-operational" (pce, igp)
  BIFT-ID: <SD subdomain,BSL bitstring length,SI Set Identifier>
  BIFT-Name: string (optional)
  BFR-ID: 16 bit (BIER-TE ID of the <bfr> in this BIFT
               or undefined if not BFER in this BIFT)
  Ingres-groups: (list of) string (1..16 bytes)
                 (that <bfr> is a member of)
  EF: <TBD> (optional, parameters for EF Function on this BIFT)
  OAM: <TBD> (optional, parameter for OAM Function on this BIFT)
  Bits: (#BSL - BitStringLength)
        BitIndex: 1...BSL
        BitType(/Tag): "unassigned",
                       (if unassigned, must have no adjacencies)
                       "unique", "p2p", "lan", "leaf", "node", "flood",
                       "group"
                       (more BitTypes defined in text below)
  Names: (list of 0 or more) string (1..16 bytes)
         (for BitTypes that require it)
  List of 0 or more adjacencies:
    (The following is the list of possible types of adjacencies,
     as defined in BIER-TE-ARCH with parameters)
    local_decap:
      VRFcontext: string (TBD)
    forward_connected:
      destination-id: ip-addr (4/16 bytes, router-id/link-local)
      link-id: ifIndex Value (connecting to destination)
      boolean: DNR (Do Not Reset)
    forward_routed:
      destination-id: 20 bit (SID), 4 or 16 bytes (router-id)
      TBD: path/encap information (e.g: SR SID stack)
  ECMP:
    list of 2 or more forward_connect and/or
    forward_routed adjacencies

```

Figure 2: BIER-TE topology information

The above picture shows informally the data model for BIER-TE topology information. <BFR> is a domain-wide unique identifier of a BFR, for example the router-id of the IGP (if an IGP is used). Every <BFR> has a "configured" instance of the BIFT information for every BIFT configured on it. This configuration could be created from legacy models, a YANG model, PCEP, or other means.

Every <BFR> also has an "operational" instance of the BIFT information. If the BFR has nor "learned-configured" / "learned-operational" information, then the "operational" instance is just a

copy of the "configuration" instance, but would take additional local information into account. For example, if resource limits do not allow to activate configured BIFT. Or when bits in the BIFT point to interfaces/adjacencies that are down, this could potentially also be reflected in the operational instance. While the "configuration" instance is read/write, the operational instance is read-only (from NMS or PCEC).

To calculate paths/bitstrings through the topology without the help of a PCEC, a BIFT would need to know the network wide BIER-TE topology. This topology consists of the "operational" BIFT informations of the BFR itself plus the "learned-operational" BIFT information from all other BIER-TE nodes in the network plus the underlay routing topology information, for example from an IGP. When an IGP is used, the "learned-operational" information of another BFR is simply learned because the BFRs are flooding this information as IGP information.

In the absence of any IGP, or the desire not to use it to distribute BIER-TE topology information, an NMS or PCEC could collect the "operational" BIER-TE topology information from BFRs and distribute it to BFRs to enable them to calculate BIER-TE bitstrings autonomously.

The operational instance of the topology information can depend on the presence of an IGP. If the adjacency of a bit in the BIFT is configured to use a nexthop identifier that has to be learned from an IGP, such as a Segment Routing SID or a router-ID, then the operational instance (as well as distributed learned-operational ones) would indicate that such an adjacency is non-operational if the BFR could not resolve this nexthop information. Forward_connected adjacencies do not require a routing underlay, but just link-local connectivity.

Some information elements in the BIER-TE topology information is metadata to support automatic consistency checking of learned topology information which permit to prohibit use of adjacencies that would not lead to working paths or worst case could create loops. The same information can also be used to auto-configure some adjacencies, specifically routed adjacencies, allowing to minimize operator work in case BIFT topology information is not auto-created from an NMS/PCEP but through manual mechanisms, but also to automatically discover mis-wirings and avoid them to be used.

The semantic of BitType and Names are described in conjunction with consistency checking and autoconfiguration in the following sections.

2.3. Consistency checking

The BitType and associated Name or Names for the bit are intended to support automated consistency checking and different reactions. an NMS can for example discover misconfiguration or miscablings and alert the operator. BFIR can likewise discover misconfiguration when the "configured" and "operational" instances of BFR are distributed via the IGP and are therefore available as "learned-configured" and "learned-operational" on the BFIR. The BFIR can then fr example stop using those misconfigured bits in any bitstrings it calculates and further escalate (e.g.: overlay signaling) unreachability of any BFER (or inability to calculate paths supporting required TE features).

"Unique" bits doe not require a name, but the <SD,SI> bit in question must only have an adjacency on one BFR. If it shows up with adjacencies on more than one BFR, this is an inconsistency.

"p2p" bits need to be the same bit on both BFR connected to each other via a subnet, and must be pointing to each other via "forward_connected" adjacencies. A "p2p" bit needs to have one Name parameter unique in the domain - for example constructed from concatenating the IfIndex of both sides. Note that the actual subnet does not need to be p2p, a BFR can have multiple bits across a multiaccess subnet, one for each neighbor.

Not listed in the above picture, but a "remote-p2p" could be a BitType when a bidirectional adjacency between two remode BFR using forward_routed adjacencies.

A "leaf" bit is the one shared bit in a <SD,SI> bitstring assigned to the "local_decap" adjacency on all leaf BFER. Leaf BFER do not need a separate bit. See BIER-TE-ARCH. If more then one "lead" bits are used in an <SD,SI> across the domain that is an inconsistency - waste of bits.

A "node" bit is associated with a Name that follows a standardized form to identify a node - e.g.: its router-id. On a non-leaf BFER, this bit can only have one local_decap adjacency on the node indicated itself. On a leaf BFER, the "node" bit must be assigned to adjacencies on one or BFR that connect to the indicated BFER. Other configurations (or wirings) are a misconfiguration.

A "lan" bit indicates a bit for a LAN, as discussed in BIER-TE-ARCH. It must have one domain wide unique name. It must only be used by BFR connecting to the same subnet with a set of forward_connected adjacencies pointing to the other BFR on that subnet. Disabling the use of a "lan" bit either on a BFIR when sending packets, or even more son on the actual BFR connecting to a subnet and recognizing

inconsistent BIER-TE topology configuraiton for it - is the most important automatic function to avoid mis-routing of BIER-TE packets. The looping will be also stopped because bits are reset when packets traverse the paths, or ultimately by TTL, but neither mechanism can provide as specifica OAM information about what went wrong than recognizing inconsistencies via the IGP.

TBD: flood bit, DNR (like lan bit, but more complex.

Consistency checking may happen directly during configuration as well as later during rewiring/remot changes of topology.

In general, the operational instance of the BIER-TE topology are relevant to topology consistency checking (as hey are for path calculations). For example, future extensions may actually introduce some form of node/BFR redundancy where different BFR are configured for the same bits, but only one at a time is actively using a bit, and therefore announcing it in the operational instance of the BIER-TE topology.

2.4. Auto-configuration

For subnets, the actual adjacency to the neighbor on a link may not actually be configured explicitly, but only the interface. Discovery of the neighbor via the IGP would result in a complete working adjacency for a bit, and that adjacency would show then in the operational instance - while the configured instance would only show an incomplete adjacency and the bit that was configured for the adjacency. The Name parameter can be used in configuration to lock in the BFR that is expected to be on the other side of a subnet interface. If that node is not the one actually connected, the adjacency in the operational instance would not be completed.

When a "p2p" BitType is used, but the bit is configured inconsistently on both sides of a p2p link, an autoconfiguration mechanism may be specified to select which of the two bits should be used (e.g.: bit number configured on the higher router-id peer). This could help to auto-correct a configuration mistake, but it does of course not recover the inconsistently configured bit directly, it just ignores it.

When a "lan" or "flood" BitType is configured, likewise auto-configuration can be done to overcome misconfigurations. TBD: more details.

Most importantly, configuration of routed adjacencies can create most need for network-wide consistent configuration. This can be automated with the proposed "group" bitttype.

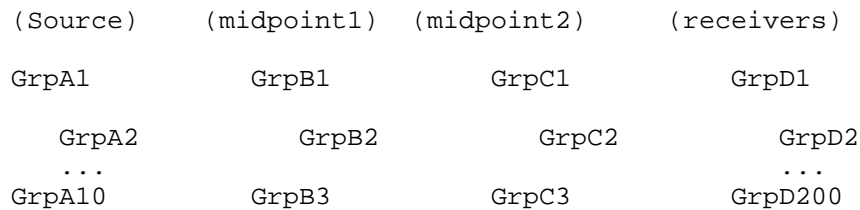


Figure 3: Group BitType use

The typical set of forward_routed adjacency is to allow steering of BIER-TE packets through a sequence of one or more members of a hop-group, load-balancing across them for TE reasons. In the above picture, those paths would start from a BFIR in GrpA and go via one (or more) nodes in GrpB, then GrpC and then BFER (GrpD).

To half-automate the setup of such loose hops, each member of GrpC would for example be configured with one unique bit of BitType "group" and the Name parameter would be set to "GrpB". Each midpoint1 BFR would "GrpB" in the list of strings for the BIFT Ingres-Group parameter. When such a BFR discovers (e.g.: via the IGP) a BFR "learned-operational" bit of BitType group with a name "GrpB" (and no adjacency!), then that midpoint1 BFR would create an adjacency in its "operational" instance, pointing to the announcing BFR with a "forward_routed" adjacency.

The saving through such group BitTypes is therefore that the bit had only to be configured on one node (the receiver side of the forward_routed adjacency), but would be configured on any number of ingres BFR for the adjacency. In the above picture, the benefit would be biggest if forward_routed adjacencies were used from Source to midpoint1, because the number of Sources is potentially largest (e.g: as shown in the picture 10 BFIR in Source group).

3. Flow Management

3.1. Operational / Architectural Models

Once a BIER-topology is active in a network, it can be used to pass BIER-TE packets. Typically this also requires the provisioning of some routing overlay because today, all applications defined for BIER today are classical SP PE-PE application where some customer traffic is mapped to SP traffic via PE-PE "overlay" signaling.

Applications in future environments such as industrial control or IoT may result in different overlay signaling. Even native end-to-end BIER-TE from application stacks is possible, but has so far not been defined.

Overlay signaling is currently out of scope of this document.

3.1.1. Overprovisioning

In the "overprovisioning flow management" model, the network operator is responsible to engineer the available network resources, BIER-TE Topology and applications generating BIER-TE flows such that the required resources can be guaranteed without contention - and potentially without the help of either PCEP or IGP, but simply using provisioning to configure BFIR and overlay signaling to determine active destinations.

Overprovisioning is the most control/signaling lightweight approach and currently the standard approach in most enterprises and service provider for IP multicast traffic.

For example: An ISP with a ++40Gbps network and a comparable small amount of high-value muticast traffic requiring in aggregate less than 5 Gbps can easily carry all of that multicast traffic across any available path. This is especially easy when the majority of traffic is best effort traffic (such as Internet traffic). In that case, the multicast traffic would be carried in a traffic class that is overprovisioned, for example with 6 Gbps guaranteed on every link. Calculated BIER-TE bitstrings would for example be used to reduce cost of multicast distribution (e.g.: steiner tree calculation), use disjoint paths (in conjunction with EF), or simply load-balance across all available non-ECMP paths. Overprovisioning flow management is traditional in most SP networks (core/edge/access) for IP multicast traffic and requires no additional signaling.

The overprovisioning flow management model is one that likely would request for (only) a YANG model to provision the BIER-TE topology.

3.1.2. PCEC

In the PCEC based flow management model, a PCEP determines (calculates) the (flow-id,<SD,SI>,bitstring) for a traffic flows and signals this to the BFIR sourcing the flow (its BFR-ID is part of the flow-id). If the flow was not statically defined, then this step would be preceeded with the BFIR requesting the resources for the, indicating the requested resources as well as the set of destinations. The destinations could be indicated as BFR-ID or (likely easier for the BFIR) by their unique identifiers in unicast routing (e.g.: router-ID). The bitstring returned by the PCEP would include not only engineered paths to all these destinations, but those paths could also be disjoint paths, carrying the traffic twice towards each destination and merging them via the EF function. The BFIR could be fully agnostic to these PCEP choices.

One of the core benefits of using BIER-TE forwarding is the ability to change the bitstring on a per-packet basis to re-route traffic by setting different transit bits, or to quickly add/delete destinations. When the BFIR should be empowered to perform any of these functions without the need for help by the PCEP, then the PCEP needs to provide additional information back to the BFIR.

If a BFIR has for example an OAM capability to determine without the help of a controller that a path has failed (too much packet loss on destination, signalled back to BFIR), and dual-transmission is not desired (due to double resource usage), then the PCP and BFIR could co-operate on a path-protection scheme in which the PCEP provides for flows not one, but two bitstrings, one being the backup path which is used by the BFIR when it discovers via OAM loss on the currently used path. This approach can extremely reduce the need to rely on controller help during failures.

When the destinations for a particular flow can potentially change over time, this can often be faster and more efficiently signalled directly via the overlay signaling to the BFIR instead of going through the PCEP. To support this mode of operations, the BFIR could request from the PCEP not simply the current set of destinations for a flow, but instead the maximum superset of receivers and request per-destination information. The PCEP would then return not just one bitstring, but one bitstring per destination (BFER). The BFIR would simply OR the bitstrings for all required destinations for each packet to create the final bitstring for that packet. Note that this description is of course on a per- $\langle SD, SI \rangle$ (aka: per BIFT) basis. Destinations using different BIFTs require always different BIER-TE packets to be sent by the BFIR.

3.1.2.1. per-flow QoS - policer/shaper/EF

In the PCEP based resource management model, it is up to the PCEP to determine how explicit resource reservations should be managed, e.g.: whether or how it tracks resource consumption. The BIER-TE forwarding plane itself does not support per-flow state with the exception of EF, which would usually be a function enabled on BFER.

Likewise, per-flow policer and/or shaper state may be a useful optional feature that the PCEP should be able to request to be enabled on a BFIR to ensure that the traffic passed by the BFIR into the BIER-TE domain does not overrun resources available. In the simplest case, such a shaper/policer could simply reflect the resources indicated by the BFIR in its request to the PCEP.

Per-flow policer/shaper or EF may need to be explicitly instantiated by BFIR/BFER. Instantiation of the Policier/Shaper on the BFIR can

happen as a function of the PCEP signaling to the BFIR, but instantiation of the EF would also require signaling of the PCEP to the BFER(s) for flows. Note that EF could also be instantiated on any midpoint BFR, so the PCEP would need to know the BIER-TE topology including where EF is considered and manage it through appropriate signaling.

Note that it is unclear yet, if EF implementations could or should be implemented with or without the need for explicit instantiation, the BIER-TE-EF-OAM document allows both options. Even in the absence of explicit signaling, per-flow Policer/Shaper and EF are limited resources and PCEP should keep track of how much of these resources are allocated and available for future flows. Like other path resources, exhaustion may require PCEP failure to allocate responses or other mitigating options.

3.1.2.2. DiffServ QoS

The only resource management that could be expected to exist in the BIER-TE domain hop-by-hop would be DiffServ QoS. As outlined in the above overprovisioning resource management model, it can serve as an easy method for lightweight resource management, and as soon as the network intends to use more than one such DiffServ codepoint across different BIER-TE flows, the PCEP should likely be able to understand and manage the DiffServ assignments of BIER-TE flows and signal the selected codepoint back to the BFIR.

3.2. BIER-TE flow model

```

BIER-TE traffic flow (change) request (from BFIR):
  Flow-control-ID: <identifier>
  Ingres BFIR of flow: (IGP router-id ?!)
  Destination-ID: set of BFER identifiers (IGP router-id ?!)
  extended-reply-required (boolean)
  Requirements:
    TSPEC (bandwidth, burst size,...)
    resilience: dual-transmission with EF
    shared-group: name

BIER-TE traffic flow reply/command (to BFIR):
  Flow-control-ID: <identifier>
  Ingres Policer/Shaper parameters (applies to each BIFT)
  Set of 1 or more BIFT:
    <SD, SI, BSL>
    BFIR-ID, entropy  (form together flow-ID)
    Bitstring
    QoS, TTL,

BIER-TE traffic flow extended reply/command (to BFER):
  Flow-control-ID: <identifier>
  Ingres Policer/Shaper parameters (applies to each BIFT)
  Set of 1 or more BIFT:
    <SD, SI, BSL>
    BFIR-ID, entropy  (form together flow-ID)
    QoS, TTL
  List of 1 or more destinations
    Destination-ID, Bitstring

BIER-TE traffic flow command (to BFER):
  Flow-control-ID: <identifier>
  Ingres BFIR of flow: BFIR-ID (in BIER-TE packet header)
  Set of 1 or more BIFT:
    <SD, SI, BSL>
    BFIR-ID, entropy  (form together flow-ID)
    EF parameter (window size etc..)

```

Figure 4: Flow request/reply/commands

The above picture shows an initial abstract representation of the data models for the different type of request/replies discussed in the previous section between PCEC and BFIR (and in one case BFER).

The Flow-control-ID identifies the managed object itself: a flow to be sent from one BFIR to a set of BFER with some TE requirements, which ultimately may require BIER-TE packets for one or more BIFT.

BFIR and BFER need to be identified in the request in a form not specific to the bits of BIFT, so the PCEP can select the appropriate BIFT(s) to use. The above picture assumes the router-id of BFIR and BFER are appropriate.

The request includes TE requirements, including (something like a) TSPEC for bandwidth, burst-size or the like, whether or not dual-transmission via PREF is required, and if the resource used are to be shared across multiple flows, then the name of a shared group. One example of sharing would for example be a video-conference where the speaker transmits video, every speaker requests/allocates a BIER-TE flow from the PCEP, but the resources for those flows are of course shared (only one flow active at a time).

The reply from the PCEP lists the BIFTS/packets that must be sent by the BFIR to reach the desired destinations as well as any other BIER-TE packet header fields relevant <SD,SI,BSL>, BFIR-ID, entropy, QoS, TTL. Beside the BIER-TE packet header, the parameters for the policer and/or shaper to be used by the BFIR are signalled back.

The extended reply does not provide simply the bitstring to use for each BIFT, but instead lists the bitstrings required for each destination so that (as described above), the BFIR can simply add/delete destinations on a packet-by-packet basis OR'ing those bitstrings.

Finally, a command to BFER is required to instruct the creation of EF state in case this can not be done automatically.

4. Security Considerations

TBD.

5. IANA Considerations

This document requests no action by IANA.

6. Acknowledgements

TBD.

7. Change log [RFC Editor: Please remove]

00: Initial version.

8. References

- [I-D.eckert-bier-te-frr]
Eckert, T., Cauchie, G., Braun, W., and M. Menth,
"Protection Methods for BIER-TE", draft-eckert-bier-te-
frr-02 (work in progress), June 2017.
- [I-D.huang-bier-te-encapsulation]
Huang, R., Eckert, T., Wei, N., and P. Thubert,
"Encapsulation for BIER-TE", draft-huang-bier-te-
encapsulation-00 (work in progress), March 2018.
- [I-D.ietf-bier-bier-yang]
Chen, R., hu, f., Zhang, Z., dai.xianxian@zte.com.cn, d.,
and M. Sivakumar, "YANG Data Model for BIER Protocol",
draft-ietf-bier-bier-yang-03 (work in progress), February
2018.
- [I-D.ietf-bier-isis-extensions]
Ginsberg, L., Przygienda, T., Aldrin, S., and Z. Zhang,
"BIER support via ISIS", draft-ietf-bier-isis-
extensions-09 (work in progress), February 2018.
- [I-D.ietf-bier-ospf-bier-extensions]
Psenak, P., Kumar, N., Wijnands, I., Dolganow, A.,
Przygienda, T., Zhang, Z., and S. Aldrin, "OSPF Extensions
for BIER", draft-ietf-bier-ospf-bier-extensions-15 (work
in progress), February 2018.
- [I-D.ietf-bier-te-arch]
Eckert, T., Cauchie, G., Braun, W., and M. Menth, "Traffic
Engineering for Bit Index Explicit Replication (BIER-TE)",
draft-ietf-bier-te-arch-00 (work in progress), January
2018.
- [I-D.thubert-bier-replication-elimination]
Thubert, P., Eckert, T., Brodard, Z., and H. Jiang, "BIER-
TE extensions for Packet Replication and Elimination
Function (PREF) and OAM", draft-thubert-bier-replication-
elimination-03 (work in progress), March 2018.
- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A.,
Przygienda, T., and S. Aldrin, "Multicast Using Bit Index
Explicit Replication (BIER)", RFC 8279,
DOI 10.17487/RFC8279, November 2017,
<<https://www.rfc-editor.org/info/rfc8279>>.

[RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <<https://www.rfc-editor.org/info/rfc8296>>.

Author's Address

Toerless Eckert
Futurewei Technologies Inc.
2330 Central Expy
Santa Clara 95050
USA

Email: tte+ietf@cs.fau.de

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: September 6, 2018

R. Huang
T. Eckert
N. Wei
Huawei
P. Thubert
Cisco
March 5, 2018

Encapsulation for BIER-TE
draft-huang-bier-te-encapsulation-00

Abstract

This document proposes an enhanced encapsulation for BIER to support BIER, BIER-TE and a control word.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Terminology	2
2. BIER-TE Encapsulation (normative)	3
2.1. BT bit - Simultaneous support for BIER and BIER-TE	3
2.2. BIFT-ID	3
2.3. Control Word and flows	3
2.4. Header format & fields	4
3. BIER-TE based resilience operations (informational)	5
4. BitStringLength (BSL) considerations (informational)	6
4.1. IPTV	7
4.2. Multicast in L3VPN	8
5. Acknowledgements	9
6. Security Considerations	9
7. IANA Considerations	9
8. References	9
8.1. Normative References	9
8.2. Informative References	10
Authors' Addresses	10

1. Introduction

[BIER-TE-ARCH] specifies BIER-TE: Traffic Engineering for Bit Index Explicit Replication (BIER). It builds on the BIER architecture as described in RFC8279 [RFC8279], but uses every BitPosition of the BitString of a BIER-TE packet to indicate one or more adjacencies instead of a BFER as in BIER.

This document proposes an enhanced version of the MPLS and non-MPLS encapsulation for BIER packets to support both BIER and BIER-TE. It is based on RFC8296 [RFC8296].

This enhanced encapsulation also adds support for a control word in the header and discusses it. Finally, the document discusses BitStringLength (BSL) size requirements in implementations for informational reasons to help aid implementors to determine an appropriate BSL.

1.1. Terminology

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 RFC2119 [RFC2119].

2. BIER-TE Encapsulation (normative)

2.1. BT bit - Simultaneous support for BIER and BIER-TE

This document supports mixed BIER and BIER-TE forwarding in a domain. Either or both of them may be used in a domain. The overall solution to support this depends on additional signaling such as existing BIER ISIS/BGP signaling. Architecturally, every SD SHOULD only use a single Type of BIER: BIER or BIER-TE. Note that this document will use the abbreviation BT to refer to the Bier Type.

In the presence of BIER and BIER-TE together in the network, there is always a risk of receiving a packet which is meant to be of one BT and processing it through a BIFT of the other BT. This can come from misconfiguration even in the face of signalling via IGP/BGP. The risk increases also when packets are generated modular from applications on PE or other sources and could use both BTs. To resolve this, the header includes a bit to indicate the BT. If the BT of a packet is inconsistent with the BT of the BIFT on the BFR, the BFR MUST NOT forward it. OAM actions MAY be triggered (subject to future work).

Note that the TTL field of the existing BIER packet header (or of IP packets) spends 7 bits on loop prevention. One bit for the BT is a comparably low cost to protect against a similar degree of problems.

Indicating the BT explicitly through a bit in the encapsulation is called the "explicit" option. Relying solely on the BT of the BIFTs is called "implicit" option. In this version of the document, we choose the explicit option for reasons outlined above.

2.2. BIFT-ID

Like in the original BIER header, the semantic of the BIFT-id of the header is that it is representing a <SI,SD,BSL> on the BFR receiving the packet. In the case of MPLS forwarding, the expectation is that the protocols to signal label ranges would be extended to also signal label ranges for the SD using BIER-TE. This is subject to the work of other documents. In the case of non-MPLS forwarding, no additional signaling may be necessary, and BIER and BIER-TE packets using the encapsulation of this document would equally use the BIFT-ID encoding as described in [BIER-non-MPLS].

2.3. Control Word and flows

This document adds a "control word" to the BIER packet header to allow that BIER or BIER-TE packets with this header could be used as

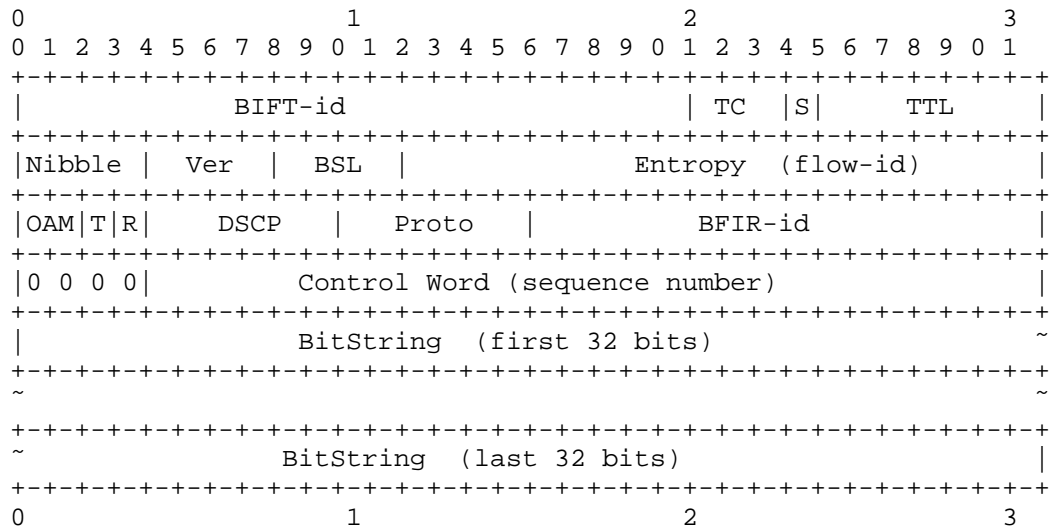
a DetNet Data Plane, independent of MPLS encapsulation, see [I-D.ietf-detnet-dp-sol], section 5.3 (in revision 01).

The control word provides a sequence number, therefore allowing to correct reordering and discover packet loss. The primary use though is resilient dual-path transmission of two copies of the same packet via disjoint paths. This is specifically a desirable use-case with BIER-TE because it allows the engineering of such disjoint paths. The flow to which the sequence number in the control word applies is <BFR-id,entropy>.

Note: The justification to carry a control word in the BIER encapsulation is similar to carrying the BFIR-ID in it. Initially, both could be seen as primarily required on BIER domain edge-nodes as part of the overlay using BIER, but not by BIER/BIER-TE itself directly. See Section 3 for more explanation how the resilience mechanism requiring the control word would work. Compared to the BFIR-ID, there is also the option to leverage it within BIER-TE itself. The details of that operations is subject to other specifications.

The authors think that the overhead of the control word is always acceptable for BIER-TE. For BIER, the use of this extended header version is optional, therefore BIER packets that need a control word would use this version of the header, those that do not need it would use version 1. If this overhead is considered to not be acceptable for all BIER-TE packets, the encoding could make those 32 bits optional through the use of one of the reserved bits or version numbers or by using a bit in the header to indicate whether the control word is present or not.

2.4. Header format & fields



All header fields not described below are left unchanged from [BIER-TE-ARCH]

T: This 1-bit field identifies that the packet is to be forwarded as a BIER packet (0) or a BIER-TE packet(1).

Ver: The version of this header format is 2.

R: Reserved - unchanged (just reduced by one bit from version). Must be set to 0.

Entropy: Unchanged, but double-used as part of the flow-identifier together with the control word

Control Word: The control word in the terminology of MPLS pseudowires (where it originates from) is the full 32 bits. For detnet, the current target is 28 bits of sequence number and 4 bits 0 preceding it.

3. BIER-TE based resilience operations (informational)

This section discusses how resilience operations with the help of the sequence number in the control word of the header in this document can be operated as an overlay (BFIR-BFER) function but also points out that it could become an integral (optional) part of BIER-TE itself. This section is solely informational. The planned document

to describe the BIER-TE forwarding aspects of resilience operations is [I-D.thubert-bier-replication-elimination].

The BFIR determines - potentially with the help of a BIER-TE Controller Host (controller) - a bitstring that forms two disjoint DAGs (Directed Acyclic Graphs) through the BIER-TE domain towards the same set of BFER. In addition, an entropy value is decided (by BFIR and/or controller) and signalled to the BFER. The BFER can therefore set up "duplicate elimination state": The BFIR increments the sequence number with every packet of the flow it sends. The BFER assigns packets to a flow by <bfir-id,entropy> and performs duplicate elimination on them.

Note that the bitstring as seen on the receiving BFER can provide additional diagnostics, for example the bits not reset by forwarding in BIER-TE give an indication about which path the BIER-TE packet was forwarded.

Instead of simply considering this protected mode of operations solely an end-to-end (BFIR/BFER) function, it could also be more flexibly embedded into BIER-TE itself, allow to provide in-BIER-TE segmented Packet Replication and (duplicate) Elimination Functions (PREF) definable by the bitstring of a BIER-TE packet. This could be achieved by adding to BIER-TE forwarding functions new adjacency types for duplication with sequence-number generation and duplicate-elimination. The ability to perform such processing as part of BIER-TE itself is the primary reason to ensure that all the necessary elements for such operations are part of the BIER-TE header itself.

4. BitStringLength (BSL) considerations (informational)

BIER-TE uses each BitPosition to indicate the adjacencies instead of a BFER as in BIER, it therefore consumes more BitPositions than BIER. In BIER-TE, the number of adjacencies passed by one BIER-TE packet MUST be less than the value of BitString length (BSL). The BIER-TE architecture discusses a range of options to reduce the number of bits for intermediate hops through various BIER-TE adjacencies and how to use them.

The maximum supported BSL has a different impact in BIER-TE than it has in BIER: A smaller maximum supported BSL in BIER primarily leads to less replication efficiency: With a BSL of 256, BIER can be up to 256 more efficient than unicast (1 packet for 256 receivers). In BIER-TE, the BSL also limits the size of the topology towards BFER and the alternative paths that can explicitly be engineered to reach the BFER. One simple guess is that 50% of bits in a bitstring may be required for intermediate hops, therefore requiring about double the amount of bits as BIER - as the cost of being able to engineer paths.

So far, there is no comprehensive analysis of the number of required bits for specific scenarios in BIER-TE. The following subsections give two examples of such scenarios and how to use and save BIER-TE bits for intermedia hops.

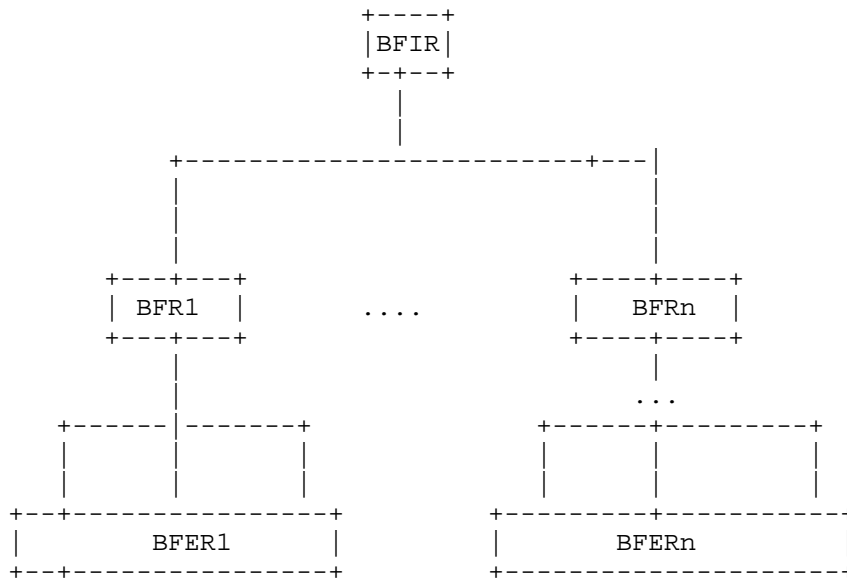
4.1. IPTV

Multicast is widely used for IPTV services by simultaneously delivering a single stream of video to thousands of recipients. Currently, PIM is widely used to provide multicast capability usually from core router(CR) to provider edges (PEs). And the multicast tree is usually constructed in the hierarchical way. The end users using PIM/IGMP to request the multicast data. BIER can be well used in from CR to those PEs. The number of hops from multicast source (CR) which could be BFIRs, to the multicast receivers (PE) which can be regarded as BRERs is usually no more than 10. BIER-TE will be useful in the cases where different video channels can have different transport paths to achieve load balancing.

To save the bit consumption, 2 ways could be used:

1. Multiple BFRs and routes are required to receive the same data. These BFRs or links can share one bit.
2. Different bits can be used for pruning. But these bits can be reused in similar but different groups.

Considering an example illustrated as following:



BFR1 and BFRn, and other BFRs in between, can share one bit because they are receiving all the content and don't need pruning. From BFR1 to BFER1, there are 3 ways to reach BFER1. So these 3 ways can be assigned with different bits. But these 3 bits can be reused in the group from BFRn to BFERn, and other groups in between, which share the same topology as the group from BFR1 to BFER1.

BIER-TE can be well implemented using these 2 ways to save the bit consumption in IPTV networks with the similar topologies like the above example.

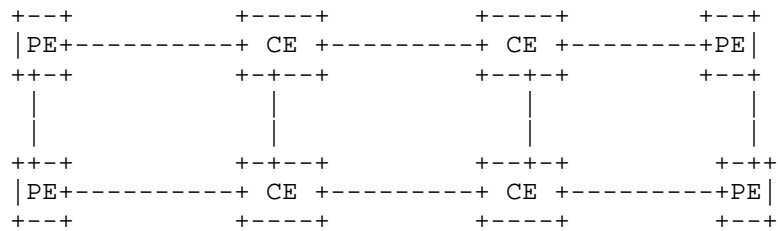
4.2. Multicast in L3VPN

MVPN is a technology to deploy multicast service in an existing VPN or as part of a transport infrastructure. Multicast data is transmitted between private networks over a VPN infrastructure by encapsulating the original multicast packets. PE routers are connected to these private networks either containing receivers or senders.

There are several multicast applications widely using the MVPN deployment. For example, L3VPN multicast service offered by service providers to enterprise customers, and video transport applications for separation between different customers: One content provider may provide video wholesale service to another, or multiple content providers may share one network to transport video from headend. Especially the latter case, network SLAs should be guaranteed as the

original video content is precious. Thus, traffic engineering is required.

According to the current implementations, the scale of a MVPN network usually contains less than several hundreds of PEs, and hundreds of core routers which are connected in full mesh, like following figure illustrated.



In such a case, the ways in Section 7.5.2 of [BIER-TE-ARCH] can be used by regarding the CE area as the Core. Based on this, current BIER design is sufficient to be reused in BIER-TE.

5. Acknowledgements

TBD.

6. Security Considerations

The security considerations are in compliance with BIER-TE architecture [BIER-TE-ARCH] and BIER encapsulation RFC8296 [RFC8296]. And the content in this document does not create any other attacks or security concerns.

7. IANA Considerations

TBD.

8. References

8.1. Normative References

- [BIER-non-MPLS]
Wijnands, I., Xu, X., and H. Bidgoli, "An Optional Encoding of the BIFT-id Field in the non-MPLS BIER Encapsulation", ID draft-wijnandsxu-bier-non-mpls-bift-encoding-01 (work in progress), August 2017.

[BIER-TE-ARCH]

Eckert, T., Cauchie, G., Braun, W., and M. Menth, "Traffic Engineering for Bit Index Explicit Replication BIER-TE", ID draft-ietf-bier-te-arch-00 (work in progress), January 2018.

[I-D.thubert-bier-replication-elimination]

Thubert, P., Eckert, T., Brodard, Z., and H. Jiang, "BIER-TE extensions for Packet Replication and Elimination Function (PREF) and OAM", draft-thubert-bier-replication-elimination-03 (work in progress), March 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.

[RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <<https://www.rfc-editor.org/info/rfc8296>>.

8.2. Informative References

[I-D.ietf-detnet-dp-sol]

Korhonen, J., Andersson, L., Jiang, Y., Finn, N., Varga, B., Farkas, J., Bernardos, C., Mizrahi, T., and L. Berger, "DetNet Data Plane Encapsulation", draft-ietf-detnet-dp-sol-01 (work in progress), January 2018.

Authors' Addresses

Rachel Huang
Huawei
101 Software Avenue, Yuhua District
Nanjing 210012
China

Email: rachel.huang@huawei.com

Toerless Eckert
Huawei USA - Futurewei Technologies Inc.
2330 Central Expy
Santa Clara 95050
USA

Email: tte+ietf@cs.fau.de

Naiwen Wei
Huawei

Email: weinaiwen@huawei.com

Pascal Thubert
Cisco Systems
Village d'Entreprises Green Side
400, Avenue de Roumanille
Batiment T3
Biot - Sophia Antipolis 06410
FRANCE

Phone: +33 4 97 23 26 34
Email: pthubert@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 26 January 2022

H. Bidgoli, Ed.
Nokia
F. Xu
Verizon
J. Kotalwar
Nokia
I. Wijnands
M. Mishra
Cisco System
Z. Zhang
Juniper Networks
25 July 2021

PIM Signaling Through BIER Core
draft-ietf-bier-pim-signaling-12

Abstract

Consider large networks deploying traditional PIM multicast service. Typically, each portion of these large networks have their own mandates and requirements. It might be desirable to deploy BIER technology in some part of these networks to replace traditional PIM services. In such cases downstream PIM states need to be signaled over the BIER Domain toward the source.

This draft specifies the procedure to signal PIM join/prune messages through a BIER Domain, as such enabling the provisioning of traditional PIM services through a BIER Domain. These procedures are valid for forwarding PIM join/prune messages to the Source (SSM) or Rendezvous Point (ASM).

Status of This Memo

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

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

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

This Internet-Draft will expire on 26 January 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Conventions used in this document	3
2.1. Definitions	3
3. PIM Signaling Through BIER domain	4
3.1. Ingress BBR procedure	4
3.1.1. New Pim Join Attribute, BIER Information Vector	5
3.1.1.1. BIER packet construction at the IBBR	6
3.2. Signaling PIM through the BIER domain procedure	7
3.3. EBBR procedure	7
4. Datapath Forwarding	7
4.1. Datapath traffic flow	8
5. PIM-SM behavior	8
6. Applicability to MVPN	8
7. IANA Considerations	9
8. Security Considerations	9
9. Acknowledgments	9
10. References	9
10.1. Normative References	9
10.2. Informative References	10
Appendix A. Determining the EBBR	10
A.1. Link-State Protocols	10
A.2. Indirect next-hop	10
A.2.1. Static Route	11
A.2.2. Interior Border Gateway Protocol (iBGP)	11
A.3. Inter-area support	11
A.3.1. Inter-area Route summarization	12
Authors' Addresses	12

1. Introduction

It might be desirable to simplify/upgrade some part of an existing network to BIER technology, removing any legacy multicast protocols like PIM. This simplification should be done with minimum interruption or disruption to the other parts of the network from singling, services and software upgrade point of view. To do so this draft specifies procedures for signaling multicast join and prune messages over the BIER domain, this draft is not trying to create FULL PIM adjacency over a BIER domain between two PIM nodes. The PIM adjacency is terminated at BIER edge routers and only join/prune signaling messages are transported over the BIER network. It just so happened that this draft chose signaling messages to be in par with PIM join/prune messages. These signaling messages are forwarded upstream toward the BIER edge router on path to the Source or Rendezvous point. These signaling messages are encapsulated in a BIER header.

2. Conventions used in this document

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.1. Definitions

An understanding of the BIER architecture [RFC8279] and the related terminology is expected. The following are some of the new definitions used in this draft.

BBR:

BIER Boundary router. A router between the PIM domain and BIER domain. Maintains PIM adjacency for all routers attached to it on the PIM domain and terminates the PIM adjacency toward the BIER domain.

IBBR:

Ingress BIER Boundary Router. An ingress router from signaling point of view. It maintains PIM adjacency toward the PIM domain and signals join/prune messages across the BIER domain to EBBR as needed.

EBBR:

Egress BIER Boundary Router. An egress router in BIER domain from signaling point of view. It maintains PIM adjacency to all upstream PIM routers. It terminates the BIER signaling packets and creates necessary PIM join/prune messages into PIM Domain.

3. PIM Signaling Through BIER domain

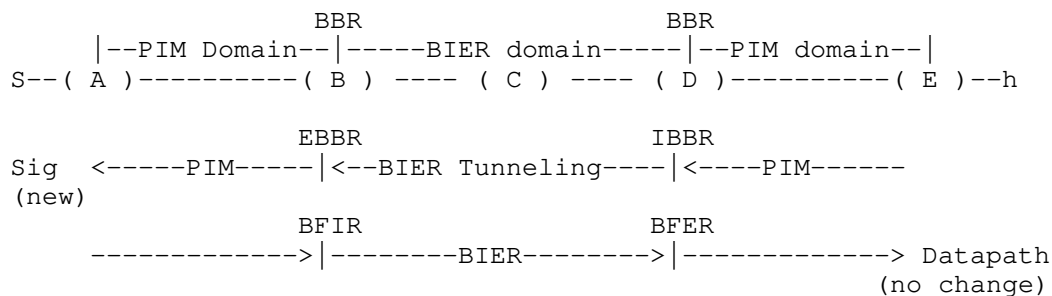


Figure 1: BIER boundary router

Figure 1 illustrates the operation of the BIER Boundary router (BBR). BBRs are connected to PIM routers in the PIM domain and BIER routers in the BIER domain. PIM routers in PIM domain continue to send PIM state messages to the BBR. The BBR will create PIM adjacency between all the PIM routers attached to it on the PIM domain. Each BBR determines if a BIER Signaling Join or Prune message needs to be transmitted through the BIER domain. This draft has chosen these BIER signaling messages to be PIM join/prune message, as such an implementation could chose to tunnel actual PIM join/prune messages through BIER network. This tunneling is only done for signaling purposes and not for creating a PIM adjacency between the two disjoint PIM domains through the BIER domain.

The terminology ingress BBR (IBBR) and egress BBR (EBBR) is relative only from a signaling point of view.

The egress BBR will determine if the arriving BIER packet is a signaling packet and if so it will generate a PIM join/prune packet toward its attached PIM domain.

The new procedures in this draft are only applicable to signaling and there are no changes from datapath point of view.

3.1. Ingress BBR procedure

The IBBR maintains a PIM adjacency [RFC7761] with any PIM router attached to it on the PIM domain.

When a PIM Join or Prune message is received, the IBBR determines whether the Source or RP is reachable through the BIER domain. The EBBR is the BFR through which the Source or RP is reachable. In PIM terms [RFC7761], the EBBR is the RPF Neighbor, and the RPF Interface is the BIER "tunnel" used to reach it. The mechanisms used to find the EBBR are outside the scope of this document and there can be many mechanism depending on if the source or RP are in same area or autonomous system (AS) or in different area or AS -- some examples are provided in Appendix A.

If the lookup for source or rendezvous point results into multiple EBBRs, different IBBRs could choose different EBBRs for the same flow. As long as a unique IBBR chooses a unique EBBR for the same flow. On downstream these EBBRs will send traffic to their corresponding IBBRs.

After discovering the EBBR and its BFR-id, the IBBR MUST use the BIER Information Vector (Section 3.1.1) which is a PIM Join Attribute type [RFC5384]. The EBBR uses this attribute to obtain the necessary BIER information to build its multicast state. The signaling packet, in this case a PIM Join/Prune message, is encapsulated in the BIER Header and forwarded through the BIER domain to the EBBR. The source address of the PIM packets MUST be set to IBBR local BFR-prefix. The destination address MUST be set to ALL-PIM-ROUTERS [RFC7761].

The IBBR will track all the PIM interfaces on the attached PIM domain which are interested in a certain (S,G). It creates multicast states for arriving join messages from PIM domain, with incoming interface as BIER "tunnel" interface and outgoing interface as the PIM domain interface(s) on which PIM Join(s) were received on.

3.1.1. New Pim Join Attribute, BIER Information Vector

The new PIM Join Attribute " BIER Information Vector" is defined as follow based on [RFC5384]

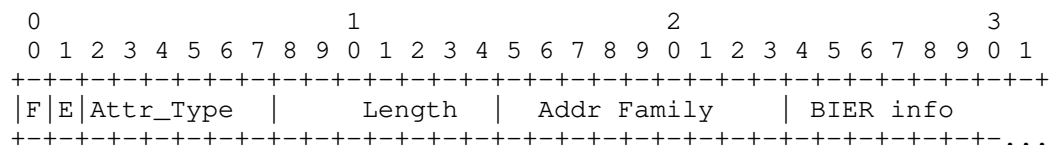


Figure 2: PIM Join Attribute

F bit: Transitive Attribute, as specified in [RFC5384]. MUST be set to zero as this attribute is always non-transitive. If EBBR receives this attribute type with the F bit set it must discard the Attribute.

E bit: End of Attributes, as specified in [RFC5384]

Attr_Type: TBD assign by IANA.

Length: Length of the value field, as specified in [RFC5384]. MUST be set to the length of the BIER Info field + 1. For IPv4 the length is 8, and 20 for IPv6. Incorrect length value compare to the Addr Family must be discarded.

Addr Family: PIM address family as specified in [RFC7761].
Unrecognized Address Family must be discarded.

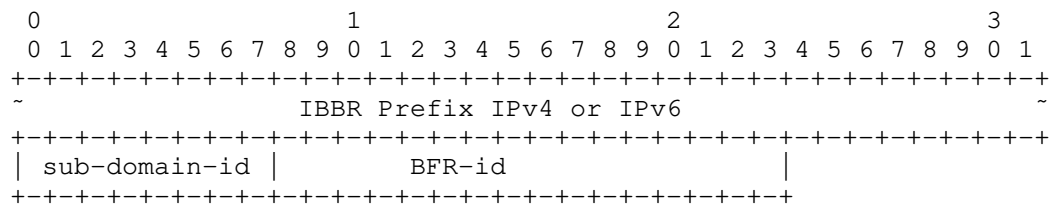


Figure 3: PIM Join Attribute detail

BIER Info: IBBR's BFR-prefix (IPv4 or IPv6), sub-domain-id, BFR-id

3.1.1.1. BIER packet construction at the IBBR

The BIER header will be encoded with the BFR-id of the IBBR (with appropriate bit set in the BitString) and the PIM signaling packet is then encapsulated in the packet.

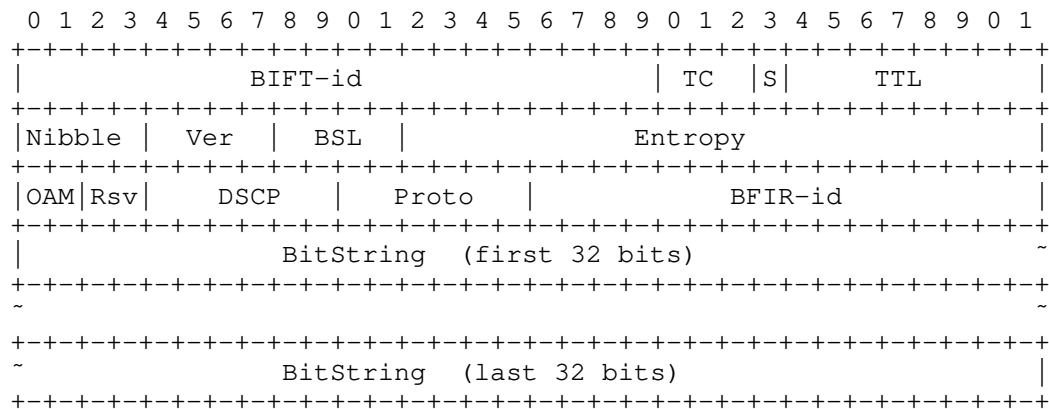


Figure 4: BIER header

BIERHeader.Proto = PIM Address Family

BIERHeader.BitString= Bit corresponding to the BFR-id of the EBBR

BIERHeader.BFIR-id = BFR-Id of the BBR originating the encapsulated signaling packet, i.e. the IBBR.

Rest of the values in the BIER header are determined based on the network (MPLS/non-MPLS), capabilities (BSL), and network configuration.

3.2. Signaling PIM through the BIER domain procedure

Throughout the BIER domain the BIER forwarding procedure is according to [RFC8279]. No BIER router will examine the BIER the signaling packet. As such there is no multicast state built in the BIER domain.

The packet will be forwarded through the BIER domain until it reaches the EBBR indicated by the BIERHeader.Bitstring. Only this targeted EBBR router will remove the BIER header and examine the PIM IPv4 or IPv6 signaling packet further as per EBBR Procedure section.

3.3. EBBR procedure

EBBR removes the BIER Header and determine this is a signaling packet. The Received signaling packet, PIM join/prune message, is processed as if it were received from neighbors on a virtual interface, (i.e. as if the pim adjacency was present, regardless of the fact that there is no adjacency).

The EBBR will build a forwarding table for the arriving (S,G) using the obtained BFIR-id and the Sub-Domain information from BIER Header and/or the PIM join Attributes added to the signaling packet. In short it tracks all IBBRs interested in this (S,G). For a specific Source and Group, EBBR SHOULD track all the interested IBBRs via signaling messages arriving from the BIER Domain. BFER builds its (s,g) forwarding state with incoming interface (IIF) as the Reverse Path Forwarding (RPF) interface (in attached PIM domain) towards the source or rendezvous point. The outgoing interfaces include a virtual interface that represent BIER forwarding to tracked IBBRs.

The EBBR maintains a PIM adjacency [RFC7761] with any PIM router attached to it on the PIM domain. At this point the end-to-end multicast traffic flow setup is complete.

4. Datapath Forwarding

4.1. Datapath traffic flow

When multicast data traffic arrives on the BFIR (EBBR) it forwards the traffic, through the BIER domain, to all interested IBBRs following the procedures specified in [RFC8279]. The BFER(s) (IBBR(s)) also follow the procedures in [RFC8279] and forward the multicast packet through its outgoing interface(s).

5. PIM-SM behavior

The procedures described in this document can be used with Any-Source Multicast (ASM) as long as a static Rendezvous Point (RP) or embedded RP for IPv6 is used [RFC3956].

It should be noted that this draft only signals PIM Joins and Prunes through the BIER domain and not any other PIM message types including PIM Hellos or Asserts. As such functionality related to these other type of messages will not be possible through a BIER domain with this draft and future drafts might cover these scenarios. As an example DR selection should be done in the PIM domain or if the PIM routers attached to IBBRs are performing DR selection there needs to be a dedicated PIM interface between these routers. The register messages are unicasts encapsulated from the source to RP as such they are forwarded without these procedures.

In case of PIM ASM Static RP or embedded RP for IPv6 the procedure for leaves joining RP is the same as above. It should be noted that for ASM, the EBBRs are determined with respect to the RP instead of the source.

6. Applicability to MVPN

With just minor changes, the above procedures apply to MVPN as well, with BFIR/BFER/EBBR/IBBR being VPN PEs. All the PIM related procedures, and the determination of EBBR happens in the context of a VRF, following procedures for PIM-MVPN.

When a PIM packet arrives from PIM domain attached to the VRF (IBBR), and it is determined that the source is reachable via the VRF through the BIER domain, a PIM signaling message is sent via BIER to the EBBR. In this case usually the PE terminating the PIM-MVPN is the EBBR. A label is imposed before the BIER header is imposed, and the "proto" field in the BIER header is set to 1 (for "MPLS packet with downstream-assigned label at top of stack"). The label is advertised by the EBBR/BFIR to associate incoming packets to its correct VRF. In many scenarios a label is already bound to the VRF loopback address on the EBBR/BFIR and it can be used.

When a multicast data packet is sent via BIER by an EBBR/BFIR, a label is imposed before the BIER packet is imposed, and the "proto" field in the BIER header is set to 1 (for "MPLS packet with downstream-assigned label at top of stack"). The label is assigned to the VPN consistently on all VRFs [draft-zzhang-bess-mvpn-evpn-aggregation-label-01].

If the more complicated label allocation scheme is needed for the data packets as specified in [draft-zzhang-bess-mvpn-evpn-aggregation-label-01], then additional PMSI signaling is needed as specified in [RFC6513].

To support per-area subdomain in this case, the ABRs would need to become VPN PEs and maintain per-VPN state so it is unlikely practical.

7. IANA Considerations

IANA is requested to assign a value (TBD) to the BIER Information Vector PIM Join Attribute from the PIM Join Attribute Types registry.

8. Security Considerations

The procedures of this document do not, in themselves, provide privacy, integrity, or authentication for the control plane or the data plane. For a discussion of the security considerations regarding the use of BIER, please see [RFC8279] and [RFC8296]. The security consideration for [RFC7761] also apply.

9. Acknowledgments

The authors would like to thank Eric Rosen, Stig Venaas for thier reviews and comments.

10. References

10.1. Normative References

- [RFC2119] "S. Brandner, "Key words for use in RFCs to Indicate Requirement Levels"", March 1997.
- [RFC5384] "A. Boers, I. Wijnands, E. Rosen, "PIM Join Attribute Format"", November 2008.
- [RFC7761] "B.Fenner, M.Handley, H. Holbrook, I. Kouvelas, R. Parekh, Z.Zhang "PIM Sparse Mode"", March 2016.

- [RFC8174] "B. Leiba, "ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words"", May 2017.
- [RFC8279] "Wijnands, IJ., Rosen, E., Dolganow, A., Przygienda, T. and S. Aldrin, "Multicast using Bit Index Explicit Replication"", October 2016.
- [RFC8296] "IJ. Wijnands, E. Rosen, A. Dolganow, J. Yantsura, S. Aldrin, I. Meilik, "Encapsulation for BIER"", January 2018.

10.2. Informative References

- [draft-zzhang-bess-mvpn-evpn-aggregation-label-01]
"Z. Zhang, E. Rosen, W. Lin, Z. Li, I. Wijnands, "MVPN/EVPN Tunnel Aggregation with Common labels"", April 2018.
- [RFC3956] "P. Savola, B. Haberman "Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address"".
- [RFC6513] "E. Rosen, R. Aggarwal, "Multicast in MPLS/BGP IP VPNs"", November 2008.

Appendix A. Determining the EBBR

This appendix provides some examples of routing procedures that can be used to determine the EBBR at the IBBR.

A.1. Link-State Protocols

On IBBR SPF procedures can be used to find the EBBR closest to the source.

Assuming the BIER domain consists of all BIER forwarding routers, SPF calculation can identify the router advertising the prefix for the source. A post process can find the EBBR by walking from the advertising router back to the IBBR in the reverse direction of shortest path tree branch until the first BFR is encountered.

A.2. Indirect next-hop

Alternatively, the route to the source could have an indirect next-hop that identifies the EBBR. These methods are explained in the following sections.

A.2.1. Static Route

A static route to the source can be configured on the IBBR with the next-hop set as the EBBR's BFR-prefix.

A.2.2. Interior Border Gateway Protocol (iBGP)

Consider the following topology:

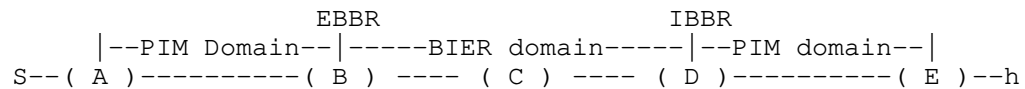


Figure 5: Static Route

Suppose BGP is enable between EBBR (B) and IBBR (D) and the PIM Domain routes are redistributed to the BIER domain via BGP, performing next-hop-self for these routes. This would include the Multicast Source IP address (S). In such case BGP should use the same next-hop as the EBBR BIER prefix. This will ensure that all PIM domain routes, including the Multicast Source IP address (S) are resolve via EBBR's BIER prefix address. When the host (h) triggers a PIM join message to IBBR (D), IBBR tries to resolve (S). It resolves (S) via BGP installed route and realizes its next-hop is EBBR (B).

A.3. Inter-area support

If each area has its own BIER sub-domain, the above procedure for post-SPF could identify one of the ABRs and the EBBR. If a sub-domain spans multiple areas, then additional procedures as described in A.2 is needed.

A.3.1. Inter-area Route summarization

In a multi-area topology, a BIER sub-domain can span a single area. Suppose this single area is constructed entirely of BIER capable routers and the ABRs are the BIER Boundary Routers attaching the BIER sub-domain in this area to PIM domains in adjacent areas. These BBRs can summarize the PIM domain routes via summary routes, as an example for OSPF, a type 3 summary LSAs can be used to advertise summary routes from a PIM domain area to the BIER area. In such scenarios the IBBR can be configured to look up the Source via IGP database and use the summary routes and its Advertising Router field to resolve the EBBR. The IBBR needs to ensure that the IGP summary route is generated by a BFR. This can be achieved by ensuring that BIER Sub-TLV exists for this route. If multiple BBRs (ABRs) have generated the same summary route the lowest Advertising Router IP can be selected or a vendor specific hashing algorithm can select the summary route from one of the BBRs.

Authors' Addresses

Hooman Bidgoli (editor)
Nokia
Ottawa
Canada

Email: hooman.bidgoli@nokia.com

Fengman Xu
Verizon
Richardson,
United States of America

Email: fengman.xu@verizon.com

Jayant Kotalwar
Nokia
Mountain View,
United States of America

Email: jayant.kotalwar@nokia.com

IJsbrand Wijnands
Cisco System
Diegem
Belgium

Email: ice@cisco.com

Mankamana Mishra
Cisco System
Milpitas,
United States of America

Email: mankamis@cisco.com

Zhaohui Zhang
Juniper Networks
Boston,
United States of America

Email: zzhang@juniper.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 27 October 2022

T.T.E. Eckert, Ed.
Futurewei
M.M. Menth
University of Tuebingen
G.C. Cauchie
KOEVOO
April 2022

Tree Engineering for Bit Index Explicit Replication (BIER-TE)
draft-ietf-bier-te-arch-13

Abstract

This memo describes per-packet stateless strict and loose path steered replication and forwarding for "Bit Index Explicit Replication" (BIER, RFC8279) packets. It is called BIER Tree Engineering (BIER-TE) and is intended to be used as the path steering mechanism for Traffic Engineering with BIER.

BIER-TE introduces a new semantic for "bit positions" (BP). They indicate adjacencies of the network topology, as opposed to (non-TE) BIER in which BPs indicate "Bit-Forwarding Egress Routers" (BFER). A BIER-TE packets BitString therefore indicates the edges of the (loop-free) tree that the packet is forwarded across by BIER-TE. BIER-TE can leverage BIER forwarding engines with little changes. Co-existence of BIER and BIER-TE forwarding in the same domain is possible, for example by using separate BIER "sub-domains" (SDs). Except for the optional routed adjacencies, BIER-TE does not require a BIER routing underlay, and can therefore operate without depending on an "Interior Gateway Routing protocol" (IGP).

Status of This Memo

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

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

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

This Internet-Draft will expire on 3 October 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Overview	3
1.1. Requirements Language	5
2. Introduction	5
2.1. Basic Examples	5
2.2. BIER-TE Topology and adjacencies	8
2.3. Relationship to BIER	9
2.4. Accelerated/Hardware forwarding comparison	11
3. Components	11
3.1. The Multicast Flow Overlay	12
3.2. The BIER-TE Control Plane	12
3.2.1. The BIER-TE Controller	14
3.2.1.1. BIER-TE Topology discovery and creation	14
3.2.1.2. Engineered Trees via BitStrings	15
3.2.1.3. Changes in the network topology	16
3.2.1.4. Link/Node Failures and Recovery	16
3.3. The BIER-TE Forwarding Plane	16
3.4. The Routing Underlay	17
3.5. Traffic Engineering Considerations	17
4. BIER-TE Forwarding	18
4.1. The BIER-TE Bit Index Forwarding Table (BIFT)	18
4.2. Adjacency Types	20
4.2.1. Forward Connected	21
4.2.2. Forward Routed	21
4.2.3. ECMP	21
4.2.4. Local Decapsulation	22
4.3. Encapsulation / Co-existence with BIER	22
4.4. BIER-TE Forwarding Pseudocode	23
4.5. BFR Requirements for BIER-TE forwarding	26
5. BIER-TE Controller Operational Considerations	27
5.1. Bit Position Assignments	27
5.1.1. P2P Links	27
5.1.2. BFER	27

5.1.3.	Leaf BFERs	27
5.1.4.	LANs	29
5.1.5.	Hub and Spoke	30
5.1.6.	Rings	30
5.1.7.	Equal Cost MultiPath (ECMP)	31
5.1.8.	Forward Routed adjacencies	34
5.1.8.1.	Reducing bit positions	34
5.1.8.2.	Supporting nodes without BIER-TE	35
5.1.9.	Reuse of bit positions (without DNC)	35
5.1.10.	Summary of BP optimizations	36
5.2.	Avoiding duplicates and loops	37
5.2.1.	Loops	38
5.2.2.	Duplicates	38
5.3.	Managing SI, sub-domains and BFR-ids	39
5.3.1.	Why SI and sub-domains	39
5.3.2.	Assigning bits for the BIER-TE topology	40
5.3.3.	Assigning BFR-id with BIER-TE	41
5.3.4.	Mapping from BFR to BitStrings with BIER-TE	42
5.3.5.	Assigning BFR-ids for BIER-TE	43
5.3.6.	Example bit allocations	43
5.3.6.1.	With BIER	43
5.3.6.2.	With BIER-TE	44
5.3.7.	Summary	45
6.	Security Considerations	46
7.	IANA Considerations	47
8.	Acknowledgements	47
9.	Change log [RFC Editor: Please remove]	48
10.	References	61
10.1.	Normative References	61
10.2.	Informative References	61
	Appendix A. BIER-TE and Segment Routing (SR)	64
	Authors' Addresses	65

1. Overview

BIER-TE is based on the (non-TE) BIER architecture, terminology and packet formats as described in [RFC8279] and [RFC8296]. This document describes BIER-TE in the expectation that the reader is familiar with these two documents.

BIER-TE introduces a new semantic for "bit positions" (BP). They indicate adjacencies of the network topology, as opposed to (non-TE) BIER in which BPs indicate "Bit-Forwarding Egress Routers" (BFER). A BIER-TE packets BitString therefore indicates the edges of the (loop-free) tree that the packet is forwarded across by BIER-TE. With BIER-TE, the "Bit Index Forwarding Table" (BIFT) of each "Bit Forwarding Router" (BFR) is only populated with BP that are adjacent to the BFR in the BIER-TE Topology. Other BPs are empty in the BIFT.

The BFR replicate and forwards BIER packets to adjacent BPs that are set in the packet. BPs are normally also cleared upon forwarding to avoid duplicates and loops.

BIER-TE can leverage BIER forwarding engines with little or no changes. It can also co-exist with BIER forwarding in the same domain, for example by using separate BIER sub-domains. Except for the optional routed adjacencies, BIER-TE does not require a BIER routing underlay, and can therefore operate without depending on an "Interior Gateway Routing protocol" (IGP).

This document is structured as follows:

- * Section 2 introduces BIER-TE with two forwarding examples, followed by an introduction of the new concepts of the BIER-TE (overlay) topology and finally a summary of the relationship between BIER and BIER-TE and a discussion of accelerated hardware forwarding.
- * Section 3 describes the components of the BIER-TE architecture, Flow overlay, BIER-TE layer with the BIER-TE control plane (including the BIER-TE controller) and BIER-TE forwarding plane, and the routing underlay.
- * Section 4 specifies the behavior of the BIER-TE forwarding plane with the different type of adjacencies and possible variations of BIER-TE forwarding pseudocode, and finally the mandatory and optional requirements.
- * Section 5 describes operational considerations for the BIER-TE controller, foremost how the BIER-TE controller can optimize the use of BP by using specific type of BIER-TE adjacencies for different type of topological situations, but also how to assign bits to avoid loops and duplicates (which in BIER-TE does not come for free), and finally how "Set Identifier" (SI), "sub-domain" (SD) and BFR-ids can be managed by a BIER-TE controller, examples and summary.
- * Appendix A concludes the technology specific sections of the document by further relating BIER-TE to Segment Routing (SR).

Note that related work, [I-D.ietf-roll-ccast] uses Bloom filters [Bloom70] to represent leaves or edges of the intended delivery tree. Bloom filters in general can support larger trees/topologies with fewer addressing bits than explicit BitStrings, but they introduce the heuristic risk of false positives and cannot clear bits in the BitString during forwarding to avoid loops. For these reasons, BIER-TE uses explicit BitStrings like BIER. The explicit BitStrings of BIER-TE can also be seen as a special type of Bloom filter, and this is how related work [ICC] describes it.

1.1. Requirements Language

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

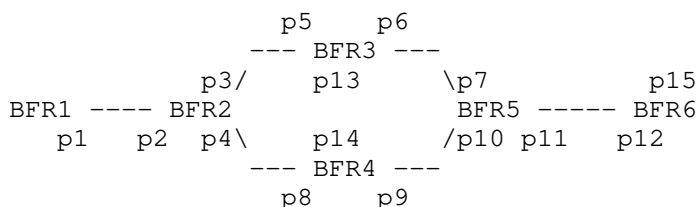
2. Introduction

2.1. Basic Examples

BIER-TE forwarding is best introduced with simple examples. These examples use formal terms defined later in the document (Figure 4), including `forward_connected()`, `forward_routed()` and `local_decap()`.

BIER-TE Topology:

Diagram:



(simplified) BIER-TE Bit Index Forwarding Tables (BIFT):

```

BFR1:  p1  -> local_decap()
       p2  -> forward_connected() to BFR2

BFR2:  p1  -> forward_connected() to BFR1
       p5  -> forward_connected() to BFR3
       p8  -> forward_connected() to BFR4

BFR3:  p3  -> forward_connected() to BFR2
       p7  -> forward_connected() to BFR5
       p13 -> local_decap()

BFR4:  p4  -> forward_connected() to BFR2
       p10 -> forward_connected() to BFR5
       p14 -> local_decap()

BFR5:  p6  -> forward_connected() to BFR3
       p9  -> forward_connected() to BFR4
       p12 -> forward_connected() to BFR6

BFR6:  p11 -> forward_connected() to BFR5
       p15 -> local_decap()

```

Figure 1: BIER-TE basic example

Consider the simple network in the above BIER-TE overview example picture with 6 BFRs. p1...p15 are the bit positions used. All BFRs can act as an ingress BFR (BFIR), BFR1, BFR3, BFR4 and BFR6 can also be BFERs. Forward_connected() is the name for adjacencies that are representing subnet adjacencies of the network. Local_decap() is the name of the adjacency to decapsulate BIER-TE packets and pass their payload to higher layer processing.

Assume a packet from BFR1 should be sent via BFR4 to BFR6. This requires a BitString (p2,p8,p10,p12,p15). When this packet is examined by BIER-TE on BFR1, the only bit position from the BitString that is also set in the BIFT is p2. This will cause BFR1 to send the only copy of the packet to BFR2. Similarly, BFR2 will forward to BFR4 because of p8, BFR4 to BFR5 because of p10 and BFR5 to BFR6 because of p12. p15 finally makes BFR6 receive and decapsulate the packet.

To send a copy to BFR6 via BFR4 and also a copy to BFR3, the BitString needs to be (p2,p5,p8,p10,p12,p13,p15). When this packet is examined by BFR2, p5 causes one copy to be sent to BFR3 and p8 one copy to BFR4. When BFR3 receives the packet, p13 will cause it to receive and decapsulate the packet.

If instead the BitString was (p2,p6,p8,p10,p12,p13,p15), the packet would be copied by BFR5 towards BFR3 because of p6 instead of being copied by BFR2 to BFR3 because of p5 in the prior case. This is showing the ability of the shown BIER-TE Topology to make the traffic pass across any possible path and be replicated where desired.

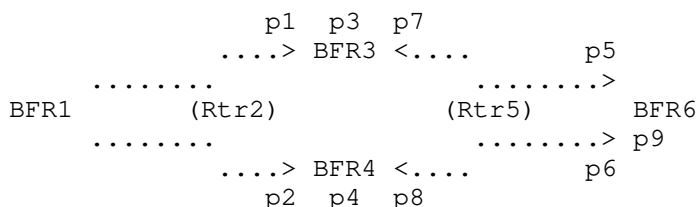
BIER-TE has various options to minimize BP assignments, many of which are based on out-of-band knowledge about the required multicast traffic paths and bandwidth consumption in the network, such as from pre-deployment planning.

Figure 2 shows a modified example, in which Rtr2 and Rtr5 are assumed not to support BIER-TE, so traffic has to be unicast encapsulated across them. To emphasize non-L2, but routed/tunneled forwarding of BIER-TE packets, these adjacencies are called "forward_routed". Otherwise, there is no difference in their processing over the aforementioned forward_connected() adjacencies.

In addition, bits are saved in the following example by assuming that BFR1 only needs to be BFIR but not BFER or transit BFR.

BIER-TE Topology:

Diagram:



(simplified) BIER-TE Bit Index Forwarding Tables (BIFT):

```

BFR1:  p1  -> forward_routed() to BFR3
       p2  -> forward_routed() to BFR4

BFR3:  p3  -> local_decap()
       p5  -> forward_routed() to BFR6

BFR4:  p4  -> local_decap()
       p6  -> forward_routed() to BFR6

BFR6:  p7  -> forward_routed() to BFR3
       p8  -> forward_routed() to BFR4
       p9  -> local_decap()
  
```

Figure 2: BIER-TE basic overlay example

To send a BIER-TE packet from BFR1 via BFR3 to be received by BFR6, the BitString is (p1,p5,p9). From BFR1 via BFR4 to be received by BFR6, the BitString is (p2,p6,p9). A packet from BFR1 to be received by BFR3,BFR4 and from BFR3 to be received by BFR6 uses (p1,p2,p3,p4,p5,p9). A packet from BFR1 to be received by BFR3,BFR4 and from BFR4 to be received by BFR6 uses (p1,p2,p3,p4,p6,p9). A packet from BFR1 to be received by BFR4, and from BFR4 to be received by BFR6 and from there to be received by BFR3 uses (p2,p3,p4,p6,p7,p9). A packet from BFR1 to be received by BFR3, and from BFR3 to be received by BFR6 there to be received by BFR4 uses (p1,p3,p4,p5,p8,p9).

2.2. BIER-TE Topology and adjacencies

The key new component in BIER-TE compared to (non-TE) BIER is the BIER-TE topology as introduced through the two examples in Section 2.1. It is used to control where replication can or should happen and how to minimize the required number of BP for adjacencies.

The BIER-TE Topology consists of the BIFTs of all the BFR and can also be expressed as a directed graph where the edges are the adjacencies between the BFRs labelled with the BP used for the adjacency. Adjacencies are naturally unidirectional. BP can be reused across multiple adjacencies as long as this does not lead to undesired duplicates or loops as explained in Section 5.2.

If the BIER-TE topology represents (a subset of) the underlying (layer 2) topology of the network as shown in the first example, this may be called a "native" BIER-TE topology. A topology consisting only of "forward_routed" adjacencies as shown in the second example may be called an "overlay" BIER-TE topology. A BIER-TE topology with both forward_connected() and forward_routed() adjacencies may be called a "hybrid" BIER-TE topology.

2.3. Relationship to BIER

BIER-TE is designed so that its forwarding plane is a simple extension to the (non-TE) BIER forwarding plane, hence allowing for it to be added to BIER deployments where it can be beneficial.

BIER-TE is also intended as an option to expand the BIER architecture into deployments where (non-TE) BIER may not be the best fit, such as statically provisioned networks with needs for path steering but without desire for distributed routing protocols.

1. BIER-TE inherits the following aspects from BIER unchanged:

1. The fundamental purpose of per-packet signaled replication and delivery via a BitString.
2. The overall architecture consisting of three layers, flow overlay, BIER(-TE) layer and routing underlay.
3. The supported encapsulations [RFC8296].
4. The semantic of all [RFC8296] header elements used by the BIER-TE forwarding plane other than the semantic of the BP in the BitString.
5. The BIER forwarding plane, except for how bits have to be cleared during replication.

2. BIER-TE has the following key changes with respect to BIER:

1. In BIER, bits in the BitString of a BIER packet header indicate a BFER and bits in the BIFT indicate the BIER control plane calculated next-hop toward that BFER. In BIER-

TE, a bit in the BitString of a BIER packet header indicates an adjacency in the BIER-TE topology, and only the BFR that is the upstream of that adjacency has its BP populated with the adjacency in its BIFT.

2. In BIER, the implied reference options for the core part of the BIER layer control plane are the BIER extensions for distributed routing protocols. This includes ISIS/OSPF extensions for BIER, [RFC8401] and [RFC8444].
 3. The reference option for the core part of the BIER-TE control plane is the BIER-TE controller. Nevertheless, both the BIER and BIER-TE BIFTs forwarding plane state could equally be populated by any mechanism.
 4. Assuming the reference options for the control plane, BIER-TE replaces in-network autonomous path calculation by explicit paths calculated by the BIER-TE controller.
3. The following elements/functions described in the BIER architecture are not required by the BIER-TE architecture:
1. "Bit Index Routing Tables" (BIRTs) are not required on BFRs for BIER-TE when using a BIER-TE controller because the controller can directly populate the BIFTs. In BIER, BIRTs are populated by the distributed routing protocol support for BIER, allowing BFRs to populate their BIFTs locally from their BIRTs. Other BIER-TE control plane or management plane options may introduce requirements for BIRTs for BIER-TE BFRs.
 2. The BIER-TE layer forwarding plane does not require BFRs to have a unique BP and therefore also no unique BFR-id. See Section 5.1.3.
 3. Identification of BFRs by the BIER-TE control plane is outside the scope of this specification. Whereas the BIER control plane uses BFR-ids in its BFR to BFR signaling, a BIER-TE controller may choose any form of identification deemed appropriate.
 4. BIER-TE forwarding does not require the BFIR-id field of the BIER packet header.
4. Co-existence of BIER and BIER-TE in the same network requires the following:

1. The BIER/BIER-TE packet header needs to allow addressing both BIER and BIER-TE BIFTs. Depending on the encapsulation option, the same SD may or may not be reusable across BIER and BIER-TE. See Section 4.3. In either case, a packet is always only forwarded end-to-end via BIER or via BIER-TE (ships in the nights forwarding).
2. BIER-TE deployments will have to assign BFR-ids to BFRs and insert them into the BFIR-id field of BIER packet headers as BIER does, whenever the deployment uses (unchanged) components developed for BIER that use BFR-id, such as multicast flow overlays or BIER layer control plane elements. See also Section 5.3.3.

2.4. Accelerated/Hardware forwarding comparison

BIER-TE forwarding rules, especially the BitString parsing are designed to be as close as possible to those of BIER in the expectation that this eases the programming of BIER-TE forwarding code and/or BIER-TE forwarding hardware on platforms supporting BIER. The pseudocode in Section 4.4 shows how existing (non-TE) BIER/BIFT forwarding can be modified to support the required BIER-TE forwarding functionality (Section 4.5), by using BIER BIFT's "Forwarding Bit Mask" (F-BM): Only the clearing of bits to avoid duplicate packets to a BFR's neighbor is skipped in BIER-TE forwarding because it is not necessary and could not be done when using BIER F-BM.

Whether to use BIER or BIER-TE forwarding is simply a choice of the mode of the BIFT indicated by the packet (BIER or BIER-TE BIFT). This is determined by the BFR configuration for the encapsulation, see Section 4.3.

3. Components

BIER-TE can be thought of being constituted from the same three layers as BIER: The "multicast flow overlay", the "BIER layer" and the "routing underlay". The following picture also shows how the "BIER layer" is constituted from the "BIER-TE forwarding plane" and the "BIER-TE control plane" represent by the "BIER-TE Controller".

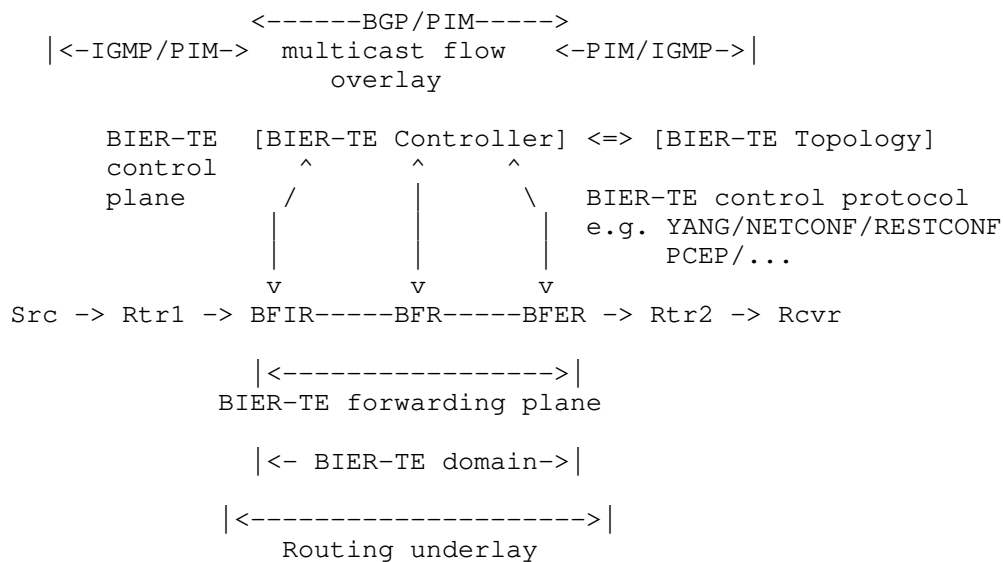


Figure 3: BIER-TE architecture

3.1. The Multicast Flow Overlay

The Multicast Flow Overlay has the same role as described for BIER in [RFC8279], Section 4.3. See also Section 3.2.1.2.

When a BIER-TE controller is used, then the signaling for the Multicast Flow Overlay may also be preferred to operate through a central point of control. For BGP based overlay flow services such as "Multicast VPN Using BIER" ([RFC8556]) this can be achieved by making the BIER-TE controller operate as a BGP Route Reflector ([RFC4456]) and combining it with signaling through BGP or a different protocol for the BIER-TE controller calculated BitStrings. See Section 3.2.1.2 and Section 5.3.4.

3.2. The BIER-TE Control Plane

In the (non-TE) BIER architecture [RFC8279], the BIER control plane is not explicitly separated from the BIER forwarding plane, but instead their functions are summarized together in Section 4.2. Example standardized options for the BIER control plane include ISIS/OSPF extensions for BIER, [RFC8401] and [RFC8444].

For BIER-TE, the control plane includes at minimum the following functionality.

1. BIER-TE topology control: During initial provisioning of the

network and/or during modifications of its topology and/or services, the protocols and/or procedures to establish BIER-TE BIFTs:

1. Determine the desired BIER-TE topology for a BIER-TE sub-domains: the native and/or overlay adjacencies that are assigned to BPs. Topology discovery is discussed in Section 3.2.1.1 and the various aspects of the BIER-TE controllers determinations about the topology are discussed throughout Section 5
 2. Determine the per-BFR BIFT from the BIER-TE topology. This is achieved by simply extracting the adjacencies of the BFR from the BIER-TE topology and populating the BFRs BIFT with them.
 3. Optionally assign BFR-ids to BFIRs for later insertion into BIER headers on BFIRs as BFIR-id. Alternatively, BFIR-id in BIER packet headers may be managed solely by the flow overlay layer and/or be unused. This is discussed in Section 5.3.3.
 4. Install/update the BIFTs into the BFRs and optionally BFR-ids into BFIRs. This is discussed in Section 3.2.1.1.
2. BIER-TE tree control: During operations of the network, protocols and/or procedures to support creation/change/removal of overlay flows on BFIRs:
1. Process the BIER-TE requirements for the multicast overlay flow: BFIR and BFERs of the flow as well as policies for the path selection of the flow. This is discussed in Section 3.5.
 2. Determine the BitStrings and optionally Entropy. This is discussed in Section 3.2.1.2, Section 3.5 and Section 5.3.4.
 3. Install state on the BFIR to impose the desired BIER packet header(s) for packets of the overlay flow. Different aspects of this and the next point are discussed throughout Section 3.2.1 and in Section 4.3, but the main responsibility of these two points is with the Multicast Flow Overlay (Section 3.1), which is architecturally inherited from BIER.
 4. Install the necessary state on the BFERs to decapsulate the BIER packet header and properly dispatch its payload.

3.2.1. The BIER-TE Controller

[RFC-Editor: the following text has three references to anchors topology-control, topology-control-1 and tree-control. Unfortunately, XMLv2 does not offer any tagging that reasonable references are generated (i had this problem already in RFCs last year. Please make sure there are useful-to-read cross-references in the RFC in these three places after you convert to XMLv3.)]

This architecture describes the BIER-TE control plane as shown in Figure 3 to consist of:

- * A BIER-TE controller.
- * BFR data-models and protocols to communicate between controller and BFRs in support of BIER-TE topology control (Section 3.2), such as YANG/NETCONF/RESTCONF ([RFC7950]/[RFC6241]/[RFC8040]).
- * BFR data-models and protocols to communicate between controller and BFIR in support of BIER-TE tree control (Section 3.2), such as BIER-TE extensions for [RFC5440].

The single, centralized BIER-TE controller is used in this document as reference option for the BIER-TE control plane but other options are equally feasible. The BIER-TE control plane could equally be implemented without automated configuration/protocols, by an operator via CLI on the BFRs. In that case, operator configured local policy on the BFIR would have to determine how to set the appropriate BIER header fields. The BIER-TE control plane could also be decentralized and/or distributed, but this document does not consider any additional protocols and/or procedures which would then be necessary to coordinate its (distributed/decentralized) entities to achieve the above described functionality.

3.2.1.1. BIER-TE Topology discovery and creation

The first item of BIER-TE topology control (Section 3.2, Paragraph 3, Item 2.2.1) includes network topology discovery and BIER-TE topology creation. The latter describes the process by which a Controller determines which routers are to be configured as BFRs and the adjacencies between them.

In statically managed networks, such as in industrial environments, both discovery and creation can be a manual/offline process.

In other networks, topology discovery may rely on protocols including extending a "Link-State-Protocol" based IGP into the BIER-TE controller itself, [RFC7752] (BGP-LS) or [RFC8345] (YANG topology) as well as BIER-TE specific methods, for example via [I-D.ietf-bier-te-yang]. These options are non-exhaustive.

Dynamic creation of the BIER-TE topology can be as easy as mapping the network topology 1:1 to the BIER-TE topology by assigning a BP for every network subnet adjacency. In larger networks, it likely involves more complex policy and optimization decisions including how to minimize the number of BPs required and how to assign BPs across different BitStrings to minimize the number of duplicate packets across links when delivering an overlay flow to BFER using different SIs/BitStrings. These topics are discussed in Section 5.

When the BIER-TE topology is determined, the BIER-TE Controller then pushes the BitPositions/adjacencies to the BIFT of the BFRs. On each BFR only those SI:BitPositions are populated that are adjacencies to other BFRs in the BIER-TE topology.

Communications between the BIER-TE Controller and BFRs for both BIER-TE topology control and BIER-TE tree control is ideally via standardized protocols and data-models such as NETCONF/RESTCONF/YANG/PCP. Vendor-specific CLI on the BFRs is also an option (as in many other SDN solutions lacking definition of standardized data models).

3.2.1.2. Engineered Trees via BitStrings

In BIER, the same set of BFER in a single sub-domain is always encoded as the same BitString. In BIER-TE, the BitString used to reach the same set of BFER in the same sub-domain can be different for different overlay flows because the BitString encodes the paths towards the BFER, so the BitStrings from different BFIR to the same set of BFER will often be different. Likewise, the BitString from the same BFIR to the same set of BFER can be different for different overlay flows for policy reasons such as shortest path trees, Steiner trees (minimum cost trees), diverse path trees for redundancy and so on.

See also [I-D.ietf-bier-multicast-http-response] for an application leveraging BIER-TE engineered trees.

3.2.1.3. Changes in the network topology

If the network topology changes (not failure based) so that adjacencies that are assigned to bit positions are no longer needed, the BIER-TE Controller can re-use those bit positions for new adjacencies. First, these bit positions need to be removed from any BFIR flow state and BFR BIFT state, then they can be repopulated, first into BIFT and then into the BFIR.

3.2.1.4. Link/Node Failures and Recovery

When link or nodes fail or recover in the topology, BIER-TE could quickly respond with FRR procedures such as [I-D.eckert-bier-te-frr], the details of which are out of scope for this document. It can also more slowly react by recalculating the BitStrings of affected multicast flows. This reaction is slower than the FRR procedure because the BIER-TE Controller needs to receive link/node up/down indications, recalculate the desired BitStrings and push them down into the BFIRs. With FRR, this is all performed locally on a BFR receiving the adjacency up/down notification.

3.3. The BIER-TE Forwarding Plane

[RFC-editor Q: "is constituted from" / "consists of" / "composed from..." ???]

The BIER-TE Forwarding Plane is constituted from the following components:

1. On a BFIR, imposition of the BIER header for packets from overlay flows. This is driven by a combination of state established by the BIER-TE control plane and/or the multicast flow overlay as explained in Section 3.1.
2. On BFRs (including BFIR and BFER), forwarding/replication of BIER packets according to their SD, SI, "BitStringLength" (BSL), BitString and optionally Entropy fields as explained in Section 4. Processing of other BIER header fields such as DSCP is outside the scope of this document.
3. On BFERs, removal of the BIER header and dispatching of the payload according to state created by the BIER-TE control plane and/or overlay layer.

When the BIER-TE Forwarding Plane receives a packet, it simply looks up the bit positions that are set in the BitString of the packet in the BIFT that was populated by the BIER-TE Controller. For every BP that is set in the BitString, and that has one or more adjacencies in

the BIFT, a copy is made according to the type of adjacencies for that BP in the BIFT. Before sending any copy, the BFR clears all BPs in the BitString of the packet for which the BFR has one or more adjacencies in the BIFT. Clearing these bits inhibits packets from looping when the BitStrings erroneously includes a forwarding loop. When a `forward_connected()` adjacency has the "DoNotClear" (DNC) flag set, then this BP is re-set for the packet copied to that adjacency. See Section 4.2.1.

3.4. The Routing Underlay

For `forward_connected()` adjacencies, BIER-TE is sending BIER packets to directly connected BIER-TE neighbors as L2 (unicasted) BIER packets without requiring a routing underlay. For `forward_routed()` adjacencies, BIER-TE forwarding encapsulates a copy of the BIER packet so that it can be delivered by the forwarding plane of the routing underlay to the routable destination address indicated in the adjacency. See Section 4.2.2 for the adjacency definition.

BIER relies on the routing underlay to calculate paths towards BFRs and derive next-hop BFR adjacencies for those paths. This commonly relies on BIER specific extensions to the routing protocols of the routing underlay but may also be established by a controller. In BIER-TE, the next-hops of a packet are determined by the BitString through the BIER-TE Controller established adjacencies on the BFR for the BPs of the BitString. There is thus no need for BFR specific routing underlay extensions to forward BIER packets with BIER-TE semantics.

Encapsulation parameters can be provisioned by the BIER-TE controller into the `forward_connected()` or `forward_routed()` adjacencies directly without relying on a routing underlay.

If the BFR intends to support FRR for BIER-TE, then the BIER-TE forwarding plane needs to receive fast adjacency up/down notifications: Link up/down or neighbor up/down, e.g. from BFD. Providing these notifications is considered to be part of the routing underlay in this document.

3.5. Traffic Engineering Considerations

Traffic Engineering ([I-D.ietf-teas-rfc3272bis]) provides performance optimization of operational IP networks while utilizing network resources economically and reliably. The key elements needed to effect TE are policy, path steering and resource management. These elements require support at the control/controller level and within the forwarding plane.

Policy decisions are made within the BIER-TE control plane, i.e., within BIER-TE Controllers. Controllers use policy when composing BitStrings and BFR BIFT state. The mapping of user/IP traffic to specific BitStrings/BIER-TE flows is made based on policy. The specific details of BIER-TE policies and how a controller uses them are out of scope of this document.

Path steering is supported via the definition of a BitString. BitStrings used in BIER-TE are composed based on policy and resource management considerations. For example, when composing BIER-TE BitStrings, a Controller must take into account the resources available at each BFR and for each BP when it is providing congestion-loss-free services such as Rate Controlled Service Disciplines [RCSD94]. Resource availability could be provided for example via routing protocol information, but may also be obtained via a BIER-TE control protocol such as NETCONF or any other protocol commonly used by a Controller to understand the resources of the network it operates on. The resource usage of the BIER-TE traffic admitted by the BIER-TE controller can be solely tracked on the BIER-TE Controller based on local accounting as long as no `forward_routed()` adjacencies are used (see Section 4.2.1 for the definition of `forward_routed()` adjacencies). When `forward_routed()` adjacencies are used, the paths selected by the underlying routing protocol need to be tracked as well.

Resource management has implications on the forwarding plane beyond the BIER-TE defined steering of packets. This includes allocation of buffers to guarantee the worst case requirements of admitted RCSD traffic and potentially policing and/or rate-shaping mechanisms, typically done via various forms of queuing. This level of resource control, while optional, is important in networks that wish to support congestion management policies to control or regulate the offered traffic to deliver different levels of service and alleviate congestion problems, or those networks that wish to control latencies experienced by specific traffic flows.

4. BIER-TE Forwarding

4.1. The BIER-TE Bit Index Forwarding Table (BIFT)

The BIER-TE BIFT is the equivalent to the BIER BIFT for (non-TE) BIER. It exists on every BFR running BIER-TE. For every BIER sub-domain (SD) in use for BIER-TE, it is a table as shown in Figure 4. That example BIFT assumes a BSL of 8 bit positions (BPs) in the packets BitString. As in [RFC8279] this BSL is purely used for the example and not a BIER/BIER-TE supported BSL (minimum BSL is 64).

A BIER-TE BIFT compares to a BIER BIFT as shown in [RFC8279] as follows.

In both BIER and BIER-TE, BIFT rows/entries are indexed in their respective BIER pseudocode ([RFC8279] Section 6.5) and BIER-TE pseudocode (Section 4.4) by the BIFT-index derived from the packets SI, BSL and the one bit position of the packets BitString (BP) addressing the BIFT row: $\text{BIFT-index} = \text{SI} * \text{BSL} + \text{BP} - 1$. BP within a BitString are numbered from 1 to BSL, hence the - 1 offset when converting to a BIFT-index. This document also uses the notion SI:BP to indicate BIFT rows, [RFC8279] uses the equivalent notion SI:BitString, where the BitString is filled with only the BP for the BIFT row.

In BIER, each BIFT-index addresses one BFER by its BFR-id = BIFT-index + 1 and is populated on each BFR with the next-hop "BFR Neighbor" (BFR-NBR) towards that BFER.

In BIER-TE, each BIFT-index and therefore SI:BP indicates one or more adjacencies between BFRs in the topology and is only populated with those adjacencies forwarding entries on the BFR that is the upstream for these adjacencies. The BIFT entry are empty on all other BFRs.

In BIER, each BIFT row also requires a "Forwarding Bit Mask" (F-BM) entry for BIER forwarding rules. In BIER-TE forwarding, F-BM is not required, but can be used when implementing BIER-TE on forwarding hardware derived from BIER forwarding, that must use F-BM. This is discussed in the first BIER-TE forwarding pseudocode in Section 4.4.

BIFT-index (SI:BP)	(FBM)	Adjacencies: <empty> or one or more per entry
BIFT indices for Packets with SI=0		
0 (0:1)	...	forward_connected(interface,neighbor{,DNC})
1 (0:2)	...	forward_connected(interface,neighbor{,DNC})
	...	forward_connected(interface,neighbor{,DNC})
...
4 (0:5)	...	local_decap({VRF})
5 (0:6)	...	forward_routed({VRF},l3-neighbor)
6 (0:7)	...	<empty>
7 (0:8)	...	ECMP((adjacency1,...adjacencyN){,seed})
BIFT indices for BitString/Packet with SI=1		
9 (1:1)
...

BIER-TE Bit Index Forwarding Table (BIFT)

Figure 4: BIER-TE BIFT with different adjacencies

The BIFT is configured for the BIER-TE data plane of a BFR by the BIER-TE Controller through an appropriate protocol and data-model. The BIFT is then used to forward packets, according to the rules specified in the BIER-TE Forwarding Procedures.

Note that a BIFT index (SI:BP) may be populated in the BIFT of more than one BFR to save BPs. See Section 5.1.6 for an example of how a BIER-TE controller could assign BPs to (logical) adjacencies shared across multiple BFRs, Section 5.1.3 for an example of assigning the same BP to different adjacencies, and Section 5.1.9 for general guidelines regarding re-use of BPs across different adjacencies.

{VRF} indicates the Virtual Routing and Forwarding context into which the BIER payload is to be delivered. This is optional and depends on the multicast flow overlay.

4.2. Adjacency Types

4.2.1. Forward Connected

A "forward_connected()" adjacency is towards a directly connected BFR neighbor using an interface address of that BFR on the connecting interface. A forward_connected() adjacency does not route packets but only L2 forwards them to the neighbor.

Packets sent to an adjacency with "DoNotClear" (DNC) set in the BIFT MUST NOT have the bit position for that adjacency cleared when the BFR creates a copy for it. The bit position will still be cleared for copies of the packet made towards other adjacencies. This can be used for example in ring topologies as explained in Section 5.1.6.

For protection against loops from misconfiguration (see Section 5.2.1), DNC is only permissible for forward_connected() adjacencies. No need or benefit of DNC for other type of adjacencies was identified and their risk was not analyzed.

4.2.2. Forward Routed

A "forward_routed()" adjacency is an adjacency towards a BFR that uses a (tunneling) encapsulation which will cause the packet to be forwarded by the routing underlay toward the adjacent BFR. This can leverage any feasible encapsulation, such as MPLS or tunneling over IP/IPv6, as long as the BIER-TE packet can be identified as a payload. This identification can either rely on the BIER/BIER-TE co-existence mechanisms described in Section 4.3, or by explicit support for a BIER-TE payload type in the tunneling encapsulation.

forward_routed() adjacencies are necessary to pass BIER-TE traffic across non BIER-TE capable routers or to minimize the number of required BP by tunneling over (BIER-TE capable) routers on which neither replication nor path-steering is desired, or simply to leverage path redundancy and FRR of the routing underlay towards the next BFR. They may also be useful to a multi-subnet adjacent BFR to leverage the routing underlay ECMP independent of BIER-TE ECMP (Section 4.2.3).

4.2.3. ECMP

(non-TE) BIER ECMP is tied to the BIER BIFT processing semantic and is therefore not directly usable with BIER-TE.

A BIER-TE "Equal Cost Multipath" (ECMP()) adjacency as shown in Figure 4 for BIFT-index 7 has a list of two or more non-ECMP adjacencies as parameters and an optional seed parameter. When a BIER-TE packet is copied onto such an ECMP() adjacency, an implementation specific so-called hash function will select one out

of the list's adjacencies to which the packet is forwarded. 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 adjacency. The seed parameter allows to design hash functions that are easy to implement at high speed without running into polarization issues across multiple consecutive ECMP hops. See Section 5.1.7 for more explanations.

4.2.4. Local Decap(sulation)

A "local_decap()" adjacency passes a copy of the payload of the BIER-TE packet to the protocol ("NextProto") within the BFR (IPv4/IPv6, Ethernet,...) responsible for that payload according to the packet header fields. A local_decap() adjacency turns the BFR into a BFER for matching packets. Local_decap() adjacencies require the BFER to support routing or switching for NextProto to determine how to further process the packet.

4.3. Encapsulation / Co-existence with BIER

Specifications for BIER-TE encapsulation are outside the scope of this document. This section gives explanations and guidelines.

Like [RFC8279], handling of "Maximum Transmission Unit" (MTU) limitations is outside the scope of this document and instead part of the BIER-TE packet encapsulation and/or flow overlay. See for example [RFC8296], Section 3. It applies equally to BIER-TE as it does to BIER.

Because a BFR needs to interpret the BitString of a BIER-TE packet differently from a (non-TE) BIER packet, it is necessary to distinguish BIER from BIER-TE packets. In the BIER encapsulation [RFC8296], the BIFT-id field of the packet indicates the BIFT of the packet. BIER and BIER-TE can therefore be run simultaneously, when the BIFT-id address space is shared across BIER BIFT and BIER-TE BIFT. Partitioning the BIFT-id address space is subject to BIER-TE/BIER control plane procedures.

When [RFC8296] is used for BIER with MPLS, BIFT-id address ranges can be dynamically allocated from MPLS label space only for the set of actually used SD:BSL BIFT. This allows to also allocate non-overlapping label ranges for BIFT-id that are to be used with BIER-TE BIFTs.

With MPLS, it is also possible to reuse the same SD space for both BIER-TE and BIER, so that the same SD has both a BIER BIFT with a corresponding range of BIFT-ids and disjoint BIER-TE BIFTs with a non-overlapping range of BIFT-ids.

When a fixed mapping from BSL, SD and SI to BIFT-id is used which does not explicitly partition the BIFT-id space between BIER and BIER-TE, such as proposed for non-MPLS forwarding with [RFC8296] encapsulation in [I-D.ietf-bier-non-mpls-bift-encoding] revision 04, section 5, then it is necessary to allocate disjoint SDs to BIER and BIER-TE BIFTs so that both can be addressed by the BIFT-ids. The encoding proposed in section 6. of the same document does not statically encode BSL or SD into the BIFT-id, but allows for a mapping, and hence could provide for the same freedom as when MPLS is being used (same or different SD for BIER/BIER-TE).

`forward_routed()` requires an encapsulation that permits to direct unicast encapsulated BIER-TE packets to a specific interface address on a target BFR. With MPLS encapsulation, this can simply be done via a label stack with that addresses label as the top label - followed by the label assigned to the (BSL,SD,SI) BitString. With non-MPLS encapsulation, some form of IP encapsulation would be required (for example IP/GRE).

The encapsulation used for `forward_routed()` adjacencies can equally support existing advanced adjacency information such as "loose source routes" via e.g. MPLS label stacks or appropriate header extensions (e.g. for IPv6).

4.4. BIER-TE Forwarding Pseudocode

The following pseudocode, Figure 5, for BIER-TE forwarding is based on the (non-TE) BIER forwarding pseudocode of [RFC8279], section 6.5 with one modification.

```
void ForwardBitMaskPacket_withTE (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;                                [3]
        BFR-NBR = BIFT[Index+Offset]->BFR-NBR;
        PacketCopy = Copy(Packet);
        PacketCopy->BitString &= F-BM;                        [2]
        PacketSend(PacketCopy, BFR-NBR);
        // The following must not be done for BIER-TE:
        // Packet->BitString &= ~F-BM;                          [1]
    }
}
```

Figure 5: BIER-TE Forwarding Pseudocode for required functions,
based on BIER Pseudocode

In step [2], the F-BM is used to clear bit(s) in PacketCopy. This step exists in both BIER and BIER-TE, but the F-BMs need to be populated differently for BIER-TE than for BIER for the desired clearing.

In BIER, multiple bits of a BitString can have the same BFR-NBR. When a received packets BitString has more than one of those bits set, the BIER replication logic has to avoid that more than one PacketCopy is sent to that BFR-NBR ([1]). Likewise, the PacketCopy sent to a BFR-NBR must clear all bits in its BitString that are not routed across BFR-NBR. This protects against BIER replication on any possible further BFR to create duplicates ([2]).

To solve both [1] and [2] for BIER, the F-BM of each bit index needs to have all bits set that this BFR wants to route across BFR-NBR. [2] clears all other bits in PacketCopy->BitString, and [1] clears those bits from Packet->BitString after the first PacketCopy.

In BIER-TE, a BFR-NBR in this pseudocode is an adjacency, forward_connected(), forward_routed() or local_decap(). There is no need for [2] to suppress duplicates in the way BIER does because in general, different BP would never have the same adjacency. If a BIER-TE controller actually finds some optimization in which this would be desirable, then the controller is also responsible to ensure that only one of those bits is set in any Packet->BitString, unless the controller explicitly wants for duplicates to be created.

The following points describe how the forwarding bit mask (F-BM) for each BP is configured in the BIFT and how this impacts the BitString of the packet being processed with that BIFT:

1. The F-BMs of all BIFT BPs without an adjacency have all their bits clear. This will cause [3] to skip further processing of such a BP.
2. All BIFT BPs with an adjacency (with DNC flag clear) have an F-BM that has only those BPs set for which this BFR does not have an adjacency. This causes [2] to clear all bits from PacketCopy->BitString for which this BFR does have an adjacency.
3. [1] is not performed for BIER-TE. All bit clearing required by BIER-TE is performed by [2].

This Forwarding Pseudocode can support the required BIER-TE forwarding functions (see Section 4.5), `forward_connected()`, `forward_routed()` and `local_decap()`, but not the recommended functions DNC flag and multiple adjacencies per bit nor the optional function, `ECMP()` adjacencies. The DNC flag cannot be supported when using only [1] to mask bits.

The modified and expanded Forwarding Pseudocode in Figure 6 specifies how to support all BIER-TE forwarding functions (required, recommended and optional):

- * This pseudocode eliminates per-bit F-BM, therefore reducing the size of BIFT state by $BSL^2 \cdot SI$ and eliminating the need for per-packet-copy BitString masking operations except for adjacencies with the DNC flag set:
 - `AdjacentBits[SI]` are bit positions with a non-empty list of adjacencies in this BFR BIFT. This can be computed whenever the BIER-TE Controller updates (add/removes) adjacencies in the BIFT.
 - The BFR needs to create packet copies for these adjacent bits when they are set in the packets BitString. This set of bits is calculated in `PktAdjacentBits`.
 - All bit positions to which the BFR creates copies have to be cleared in packet copies to avoid loops. This is done by masking the BitString of the packet with `~AdjacentBits[SI]`. When an adjacency has DNC set, this bit position is set again only for the packet copy towards that bit position.
- * BIFT entries may contain more than one adjacency in support of specific configurations such as Section 5.1.5. The code therefore includes a loop over these adjacencies.
- * The `ECMP()` adjacency is shown. Its parameters are a seed and a `ListOfAdjacencies` from which one is picked.
- * The `forward_connected()`, `forward_routed()`, `local_decap()` adjacencies are shown with their parameters.

```

void ForwardBitMaskPacket_withTE (Packet)
{
    SI = GetPacketSI(Packet);
    Offset = SI * BitStringLength;
    // Determine adjacent bits in the Packets BitString
    PktAdjacentBits = Packet->BitString & AdjacentBits[SI];

    // Clear adjacent bits in Packet header to avoid loops
    Packet->BitString &= ~AdjacentBits[SI];

    // Loop over PktAdjacentBits to create packet copies
    for (Index = GetFirstBitPosition(PktAdjacentBits); Index ;
        Index = GetNextBitPosition(PktAdjacentBits, Index)) {
        for adjacency in BIFT[Index+Offset]->Adjacencies {
            if(adjacency.type == ECMP(ListOfAdjacencies,seed) ) {
                I = ECMP_hash(sizeof(ListOfAdjacencies),
                               Packet->Entropy,seed);
                adjacency = ListOfAdjacencies[I];
            }
            PacketCopy = Copy(Packet);
            switch(adjacency.type) {
                case forward_connected(interface,neighbor,DNC):
                    if(DNC)
                        PacketCopy->BitString |= 1<<(Index-1);
                    SendToL2Unicast(PacketCopy,interface,neighbor);

                case forward_routed({VRF},l3-neighbor):
                    SendToL3(PacketCopy,{VRF},l3-neighbor);

                case local_decap({VRF},neighbor):
                    DecapBierHeader(PacketCopy);
                    PassTo(PacketCopy,{VRF},Packet->NextProto);
            }
        }
    }
}

```

Figure 6: Complete BIER-TE Forwarding Pseudocode for required, recommended and optional functions

4.5. BFR Requirements for BIER-TE forwarding

BFR that support BIER-TE and BIER MUST support configuration that enables BIER-TE instead of (non-TE) BIER forwarding rules for all BIFT of one or more BIER sub-domains. Every BP in a BIER-TE BIFT MUST support to have zero or one adjacency. BIER-TE forwarding MUST support the adjacency types `forward_connected()` with the DNC flag not set, `forward_routed()` and `local_decap()`. As explained in

Section 4.4, these required BIER-TE forwarding functions can be implemented via the same Forwarding Pseudocode as BIER forwarding except for one modification (skipping one masking with F-BM).

BIER-TE forwarding SHOULD support `forward_connected()` adjacencies with a set DNC flag, as this is highly useful to save bits in rings (see Section 5.1.6).

BIER-TE forwarding SHOULD support more than one adjacency on a bit. This allows to save bits in hub and spoke scenarios (see Section 5.1.5).

BIER-TE forwarding MAY support `ECMP()` adjacencies to save bits in ECMP scenarios, see Section 5.1.7 for an example. This is an optional requirement, because for ECMP deployments using BIER-TE one can also leverage ECMP of the routing underlay via `forwarded_routed` adjacencies and/or might prefer to have more explicit control of the path chosen via explicit BP/adjacencies for each ECMP path alternative.

5. BIER-TE Controller Operational Considerations

5.1. Bit Position Assignments

This section describes how the BIER-TE Controller can use the different BIER-TE adjacency types to define the bit positions of a BIER-TE domain.

Because the size of the BitString limits the size of the BIER-TE domain, many of the options described exist to support larger topologies with fewer bit positions.

5.1.1. P2P Links

On a P2P link that connects two BFRs, the same bit position can be used on both BFRs for the adjacency to the neighboring BFR. A P2P link requires therefore only one bit position.

5.1.2. BFER

Every non-Leaf BFER is given a unique bit position with a `local_decap()` adjacency.

5.1.3. Leaf BFERs

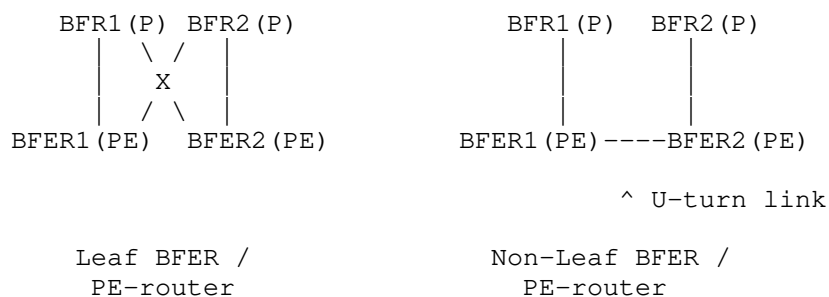


Figure 7: Leaf vs. non-Leaf BFER Example

A leaf BFER is one where incoming BIER-TE packets never need to be forwarded to another BFR but are only sent to the BFER to exit the BIER-TE domain. For example, in networks where Provider Edge (PE) router are spokes connected to Provider (P) routers, those PEs are Leaf BFERs unless there is a U-turn between two PEs.

Consider how redundant disjoint traffic can reach BFER1/BFER2 in Figure 7: When BFER1/BFER2 are Non-Leaf BFER as shown on the right-hand side, one traffic copy would be forwarded to BFER1 from BFR1, but the other one could only reach BFER1 via BFER2, which makes BFER2 a non-Leaf BFER. Likewise, BFER1 is a non-Leaf BFER when forwarding traffic to BFER2. Note that the BFERs in the left-hand picture are only guaranteed to be leaf-BFER by fitting routing configuration that prohibits transit traffic to pass through the BFERs, which is commonly applied in these topologies.

In most situations, leaf-BFER that are to be addressed via the same BitString can share a single bit position for their `local_decap()` adjacency in that BitString and therefore save bit positions. On a non-leaf BFER, a received BIER-TE packet may only need to transit the BFER or it may need to also be decapsulated. Whether or not to decapsulate the packet therefore needs to be indicated by a unique bit position populated only on the BIFT of this BFER with a `local_decap()` adjacency. On a leaf-BFER, packets never need to pass through; any packet received is therefore usually intended to be decapsulated. This can be expressed by a single, shared bit position that is populated with a `local_decap()` adjacency on all leaf-BFER addressed by the BitString.

The possible exception from this leaf-BFER bit position optimization can be cases where the bit position on the prior BIER-TE BFR (which created the packet copy for the leaf-BFER in question) is populated with multiple adjacencies as an optimization, such as in Section 5.1.4 or Section 5.1.5. With either of these two optimizations, the sender of the packet could only control explicitly

whether the packet was to be decapsulated on the leaf-BFER in question, if the leaf-BFER has a unique bit position for its `local_decap()` adjacency.

However, if the bit position is shared across leaf-BFER, and packets are therefore decapsulated potentially unnecessarily, this may still be appropriate if the decapsulated payload of the BIER-TE packet indicates whether or not the packet needs to be further processed/received. This is typically true for example if the payload is IP multicast because IP multicast on a BFER would know the membership state of the IP multicast payload and be able to discard it if the packet was delivered unnecessarily by the BIER-TE layer. If the payload has no such membership indication, and the BFIR wants to have explicit control about which BFER are to receive and decapsulate a packet, then these two optimizations can not be used together with shared bit positions optimization for leaf-BFER.

5.1.4. LANs

In a LAN, the adjacency to each neighboring BFR is given a unique bit position. The adjacency of this bit position is a `forward_connected()` adjacency towards the BFR and this bit position is populated into the BIFT of all the other BFRs on that LAN.

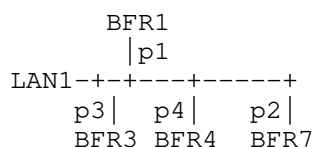


Figure 8: LAN Example

If Bandwidth on the LAN is not an issue and most BIER-TE traffic should be copied to all neighbors on a LAN, then bit positions can be saved by assigning just a single bit position to the LAN and populating the bit position of the BIFTs of each BFRs on the LAN with a list of `forward_connected()` adjacencies to all other neighbors on the LAN.

This optimization does not work in the case of BFRs redundantly connected to more than one LAN with this optimization because these BFRs would receive duplicates and forward those duplicates into the opposite LANs. Adjacencies of such BFRs into their LAN still need a separate bit position.

5.1.5. Hub and Spoke

In a setup with a hub and multiple spokes connected via separate p2p links to the hub, all p2p adjacencies from the hub to the spokes links can share the same bit position. The bit position on the hub's BIFT is set up with a list of `forward_connected()` adjacencies, one for each Spoke.

This option is similar to the bit position optimization in LANs: Redundantly connected spokes need their own bit positions, unless they are themselves Leaf-BFER.

This type of optimized BP could be used for example when all traffic is "broadcast" traffic (very dense receiver set) such as live-TV or many-to-many telemetry including situation-awareness (SA). This BP optimization can then be used to explicitly steer different traffic flows across different ECMP paths in Data-Center or broadband-aggregation networks with minimal use of BPs.

5.1.6. Rings

In L3 rings, instead of assigning a single bit position for every p2p link in the ring, it is possible to save bit positions by setting the "DoNotClear" (DNC) flag on `forward_connected()` adjacencies.

For the rings shown in Figure 9, a single bit position will suffice to forward traffic entering the ring at BFRa or BFRb all the way up to BFR1:

On BFRa, BFRb, BFR30,... BFR3, the bit position is populated with a `forward_connected()` adjacency pointing to the clockwise neighbor on the ring and with DNC set. On BFR2, the adjacency also points to the clockwise neighbor BFR1, but without DNC set.

Handling DNC this way ensures that copies forwarded from any BFR in the ring to a BFR outside the ring will not have the ring bit position set, therefore minimizing the chance to create loops.

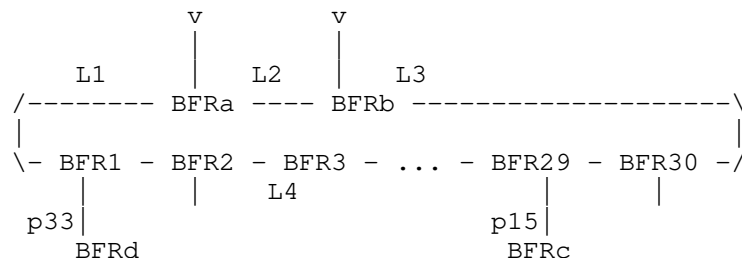


Figure 9: Ring Example

Note that this example only permits for packets intended to make it all the way around the ring to enter it at BFRa and BFRb, and that packets will always travel clockwise. If packets should be allowed to enter the ring at any ring BFR, then one would have to use two ring bit positions. One for each direction: clockwise and counterclockwise.

Both would be set up to stop rotating on the same link, e.g. L1. When the ingress ring BFR creates the clockwise copy, it will clear the counterclockwise bit position because the DNC bit only applies to the bit for which the replication is done. Likewise for the clockwise bit position for the counterclockwise copy. As a result, the ring ingress BFR will send a copy in both directions, serving BFRs on either side of the ring up to L1.

5.1.7. Equal Cost MultiPath (ECMP)

[RFC-Editor: A reviewer (Lars Eggert) noted that the infinite "to use" in the following sentence is not correct. The same was also noted for several other similar instances. The following URL seems to indicate though that this is a per-case decision, which seems undefined: <https://writingcenter.gmu.edu/guides/choosing-between-infinite-and-gerund-to-do-or-doing>. What exactly should be done about this ?].

An ECMP() adjacency allows to use just one BP to deliver packets to one of N adjacencies instead of one BP for each adjacency. In the common example case Figure 10, a link-bundle of three links L1,L2,L3 connects BFR1 and BFR2, and only one BP is used instead of three BP to deliver packets from BFR1 to BFR2.

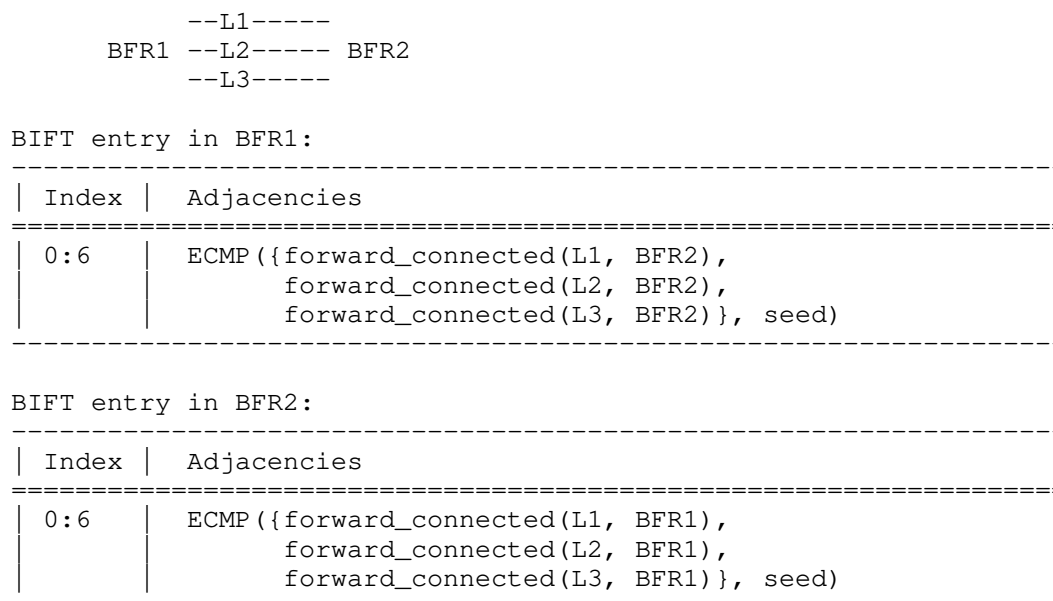


Figure 10: ECMP Example

This document does not standardize any ECMP algorithm because it is sufficient for implementations to document their freely chosen ECMP algorithm. Figure 11 shows an example ECMP algorithm, and would double as its documentation: A BIER-TE controller could determine which adjacency is chosen based on the seed and adjacencies parameters and the packet entropy.

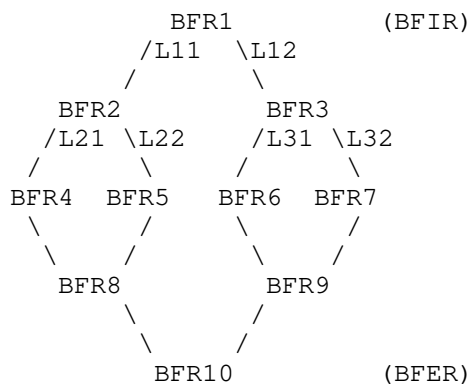
```

forward(packet, ECMP(adj(0), adj(1),... adj(N-1), seed)):
  i = (packet(bier-header-entropy) XOR seed) % N
  forward packet to adj(i)

```

Figure 11: ECMP algorithm Example

In the following example, all traffic from BFR1 towards BFR10 is intended to be ECMP load split equally across the topology. This example is not meant as a likely setup, but to illustrate that ECMP can be used to share BPs not only across link bundles, but also across alternative paths across different transit BFR, and it explains the use of the seed parameter.



BIFT entry in BFR1:

0:6	ECMP({forward_connected(L11, BFR2), forward_connected(L12, BFR3)}, seed1)
-----	--

BIFT entry in BFR2:

0:7	ECMP({forward_connected(L21, BFR4), forward_connected(L22, BFR5)}, seed1)
-----	--

BIFT entry in BFR3:

0:7	ECMP({forward_connected(L31, BFR6), forward_connected(L32, BFR7)}, seed1)
-----	--

BIFT entry in BFR4, BFR5:

0:8	forward_connected(Lxx, BFR8)	xx differs on BFR4/BFR5
-----	------------------------------	-------------------------

BIFT entry in BFR6, BFR7:

0:8	forward_connected(Lxx, BFR9)	xx differs on BFR6/BFR7
-----	------------------------------	-------------------------

BIFT entry in BFR8, BFR9:

0:9	forward_connected(Lxx, BFR10)	xx differs on BFR8/BFR9
-----	-------------------------------	-------------------------

Figure 12: Polarization Example

Note that for the following discussion of ECMP, only the BIFT ECMP adjacencies on BFR1, BFR2, BFR3 are relevant. The re-use of BP across BFR in this example is further explained in Section 5.1.9 below.

With the setup of ECMP in the topology above, traffic would not be equally load-split. Instead, links L22 and L31 would see no traffic at all: BFR2 will only see traffic from BFR1 for which the ECMP hash in BFR1 selected the first adjacency in the list of 2 adjacencies given as parameters to the ECMP. It is link L11-to-BFR2. BFR2 performs again ECMP with two adjacencies on that subset of traffic using the same seed1, and will therefore again select the first of its two adjacencies: L21-to-BFR4. And therefore L22 and BFR5 sees no traffic. Likewise for L31 and BFR6.

This issue in BFR2/BFR3 is called polarization. It results from the re-use of the same hash function across multiple consecutive hops in topologies like these. To resolve this issue, the ECMP() adjacency on BFR1 can be set up with a different seed2 than the ECMP() adjacencies on BFR2/BFR3. BFR2/BFR3 can use the same hash because packets will not sequentially pass across both of them. Therefore, they can also use the same BP 0:7.

Note that ECMP solutions outside of BIER often hide the seed by auto-selecting it from local entropy such as unique local or next-hop identifiers. Allowing the BIER-TE Controller to explicitly set the seed gives the ability for it to control same/different path selection across multiple consecutive ECMP hops.

5.1.8. Forward Routed adjacencies

5.1.8.1. Reducing bit positions

Forward_routed() adjacencies can reduce the number of bit positions required when the path steering requirement is not hop-by-hop explicit path selection, but loose-hop selection. Forward_routed() adjacencies can also allow to operate BIER-TE across intermediate hop routers that do not support BIER-TE.

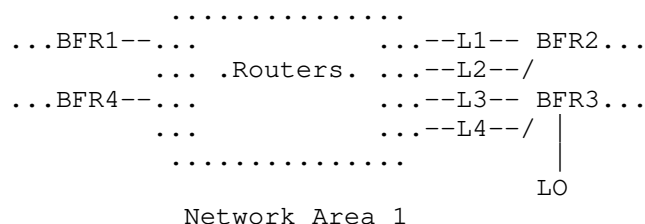


Figure 13: Forward Routed Adjacencies Example

Assume the requirement in Figure 13 is to explicitly steer traffic flows that have arrived at BFR1 or BFR4 via a path in the routing underlay "Network Area 1" to one of the following three next segments: (1) BFR2 via link L1, (2) BFR2 via link L2, or (3) via BFR3 and then not caring whether the packet is forwarded via L3 or L4.

To enable this, both BFR1 and BFR4 are set up with a `forward_routed` adjacency bit position towards an address of BFR2 on link L1, another `forward_routed()` bit position towards an address of BFR2 on link L2 and a third `forward_routed()` bit position towards a node address L0 of BFR3.

5.1.8.2. Supporting nodes without BIER-TE

`Forward_routed()` adjacencies also enable incremental deployment of BIER-TE. Only the nodes through which BIER-TE traffic needs to be steered - with or without replication - need to support BIER-TE. Where they are not directly connected to each other, `forward_routed` adjacencies are used to pass over non BIER-TE enabled nodes.

5.1.9. Reuse of bit positions (without DNC)

Bit positions can be re-used across multiple BFRs to minimize the number of BP needed. This happens when adjacencies on multiple BFRs use the DNC flag as described above, but it can also be done for non-DNC adjacencies. This section only discusses this non-DNC case.

Because BP are cleared when passing a BFR with an adjacency for that BP, reuse of BP across multiple BFRs does not introduce any problems with duplicates or loops that do not also exist when every adjacency has a unique BP. Instead, the challenge when reusing BP is whether it allows to still achieve the desired Tree Engineering goals.

BP cannot be reused across two BFRs that would need to be passed sequentially for some path: The first BFR will clear the BP, so those paths cannot be built. BP can be set across BFR that would (A) only occur across different paths or (B) across different branches of the same tree.

An example of (A) was given in Figure 12, where BP 0:7, BP 0:8 and BP 0:9 are each reused across multiple BFRs because a single packet/path would never be able to reach more than one BFR sharing the same BP.

Assume the example was changed: BFR1 has no `ECMP()` adjacency for BP 0:6, but instead BP 0:5 with `forward_connected()` to BFR2 and BP 0:6 with `forward_connected()` to BFR3. Packets with both BP 0:5 and BP

0:6 would now be able to reach both BFR2 and BFR3 and the still existing re-use of BP 0:7 between BFR2 and BFR3 is a case of (B) where reuse of BP is perfect because it does not limit the set of useful path choices:

If instead of reusing BP 0:7, BFR3 used a separate BP 0:10 for its ECMP() adjacency, no useful additional path steering options would be enabled. If duplicates at BFR10 where undesirable, this would be done by not setting BP 0:5 and BP 0:6 for the same packet. If the duplicates where desirable (e.g.: resilient transmission), the additional BP 0:10 would also not render additional value.

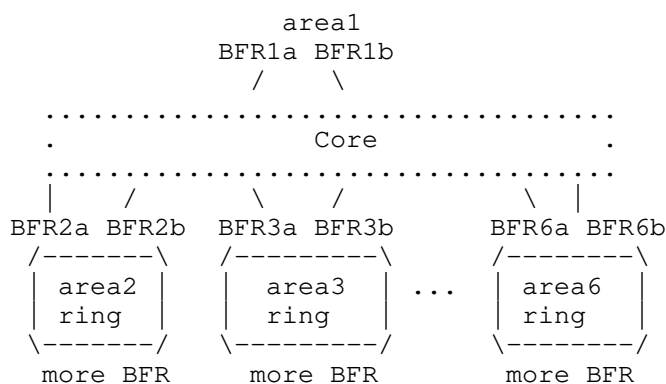


Figure 14: Reuse of BP

Reuse may also save BPs in larger topologies. Consider the topology shown in Figure 14. A BFIR/sender (e.g.: video headend) is attached to area 1, and area 2...6 contain receivers/BFER. Assume each area had a distribution ring, each with two BPs to indicate the direction (as explained before). These two BPs could be reused across the 5 areas. Packets would be replicated through other BPs for the Core to the desired subset of areas, and once a packet copy reaches the ring of the area, the two ring BPs come into play. This reuse is a case of (B), but it limits the topology choices: Packets can only flow around the same direction in the rings of all areas. This may or may not be acceptable based on the desired path steering options: If resilient transmission is the path engineering goal, then it is likely a good optimization, if the bandwidth of each ring was to be optimized separately, it would not be a good limitation.

5.1.10. Summary of BP optimizations

This section reviewed a range of techniques by which a BIER-TE Controller can create a BIER-TE topology in a way that minimizes the number of necessary BPs.

Without any optimization, a BIER-TE Controller would attempt to map the network subnet topology 1:1 into the BIER-TE topology and every subnet adjacent neighbor requires a `forward_connected()` BP and every BFER requires a `local_decap()` BP.

The optimizations described are then as follows:

- * P2P links require only one BP (Section 5.1.1).
- * All leaf-BFER can share a single `local_decap()` BP (Section 5.1.3).
- * A LAN with N BFR needs at most N BP (one for each BFR). It only needs one BP for all those BFR that are not redundantly connected to multiple LANs (Section 5.1.4).
- * A hub with p2p connections to multiple non-leaf-BFER spokes can share one BP to all spokes if traffic can be flooded to all spokes, e.g.: because of no bandwidth concerns or dense receiver sets (Section 5.1.5).
- * Rings of BFR can be built with just two BP (one for each direction) except for BFR with multiple ring connections - similar to LANs (Section 5.1.6).
- * ECMP() adjacencies to N neighbors can replace N BP with 1 BP. Multihop ECMP can avoid polarization through different seeds of the ECMP algorithm (Section 5.1.7).
- * `Forward_routed()` adjacencies allow to "tunnel" across non-BIER-TE capable routers and across BIER-TE capable routers where no traffic-steering or replications are required (Section 5.1.8).
- * BP can generally be reused across a set of nodes where it can be guaranteed that no path will ever need to traverse more than one node of the set. Depending on scenario, this may limit the feasible path steering options (Section 5.1.9).

Note that the described list of optimizations is not exhaustive. Especially when the set of required path steering choices is limited and the set of possible subsets of BFERs that should be able to receive traffic is limited, further optimizations of BP are possible. The hub and spoke optimization is a simple example of such traffic pattern dependent optimizations.

5.2. Avoiding duplicates and loops

5.2.1. Loops

Whenever BIER-TE creates a copy of a packet, the BitString of that copy will have all bit positions cleared that are associated with adjacencies on the BFR. This inhibits looping of packets. The only exception are adjacencies with DNC set.

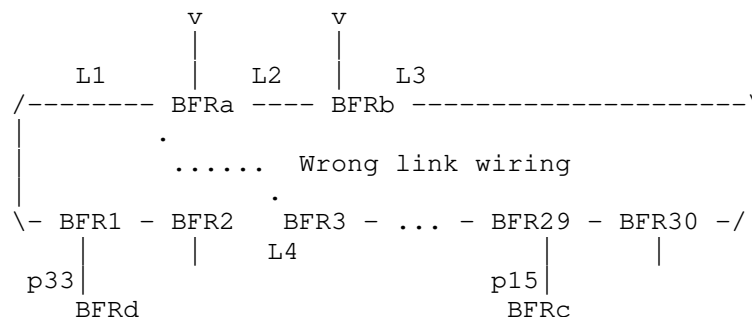


Figure 15: Miswired Ring Example

With DNC set, looping can happen. Consider in Figure 15 that link L4 from BFR3 is (inadvertently) plugged into the L1 interface of BFRa (instead of BFR2). This creates a loop where the rings clockwise bit position is never cleared for copies of the packets traveling clockwise around the ring.

To inhibit looping in the face of such physical misconfiguration, only `forward_connected()` adjacencies are permitted to have DNC set, and the link layer port unique unicast destination address of the adjacency (e.g. MAC address) protects against closing the loop. Link layers without port unique link layer addresses should not be used with the DNC flag set.

5.2.2. Duplicates

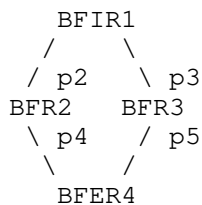


Figure 16: Duplicates Example

Duplicates happen when the graph expressed by a BitString is not a tree but redundantly connecting BFRs with each other. In Figure 16, a BitString of p2,p3,p4,p5 would result in duplicate packets to arrive on BFER4. The BIER-TE Controller must therefore ensure to only create BitStrings that are trees.

When links are incorrectly physically re-connected before the BIER-TE Controller updates BitStrings in BFIRs, duplicates can happen. Like loops, these can be inhibited by link layer addressing in `forward_connected()` adjacencies.

If interface or loopback addresses used in `forward_routed()` adjacencies are moved from one BFR to another, duplicates can equally happen. Such re-addressing operations must be coordinated with the BIER-TE Controller.

5.3. Managing SI, sub-domains and BFR-ids

When the number of bits required to represent the necessary hops in the topology and BFER exceeds the supported BitStringLength (BSL), multiple SIs and/or sub-domains must be used. This section discusses how.

BIER-TE forwarding does not require the concept of BFR-id, but routing underlay, flow overlay and BIER headers may. This section also discusses how BFR-ids can be assigned to BFIR/BFER for BIER-TE.

5.3.1. Why SI and sub-domains

For (non-TE) BIER and BIER-TE forwarding, the most important result of using multiple SI and/or sub-domains is the same: Packets that need to be sent to BFERs in different SIs or sub-domains require different BIER packets: each one with a BitString for a different (SI,sub-domain) combination. Each such BitString uses one BSL sized SI block in the BIFT of the sub-domain. We call this a BIFT:SI (block).

For BIER and BIER-TE forwarding themselves there is also no difference whether different SIs and/or sub-domains are chosen, but SI and sub-domain have different purposes in the BIER architecture shared by BIER-TE. This impacts how operators are managing them and how especially flow overlays will likely use them.

By default, every possible BFIR/BFER in a BIER network would likely be given a BFR-id in sub-domain 0 (unless there are > 64k BFIR/BFER).

If there are different flow services (or service instances) requiring replication to different subsets of BFERs, then it will likely not be possible to achieve the best replication efficiency for all of these service instances via sub-domain 0. Ideal replication efficiency for N BFER exists in a sub-domain if they are split over not more than $\text{ceiling}(N/\text{BitStringLength})$ SI.

If service instances justify additional BIER:SI state in the network, additional sub-domains will be used: BFIR/BFER are assigned BFR-id in those sub-domains and each service instance is configured to use the most appropriate sub-domain. This results in improved replication efficiency for different services.

Even if creation of sub-domains and assignment of BFR-id to BFIR/BFER in those sub-domains is automated, it is not expected that individual service instances can deal with BFER in different sub-domains. A service instance may only support configuration of a single sub-domain it should rely on.

To be able to easily reuse (and modify as little as possible) existing BIER procedures including flow-overlay and routing underlay, when BIER-TE forwarding is added, we therefore reuse SI and sub-domain logically in the same way as they are used in BIER: All necessary BFIR/BFER for a service use a single BIER-TE BIFT and are split across as many SIs as necessary (see Section 5.3.2). Different services may use different sub-domains that primarily exist to provide more efficient replication (and for BIER-TE desirable path steering) for different subsets of BFIR/BFER.

5.3.2. Assigning bits for the BIER-TE topology

In BIER, BitStrings only need to carry bits for BFERs, which leads to the model that BFR-ids map 1:1 to each bit in a BitString.

In BIER-TE, BitStrings need to carry bits to indicate not only the receiving BFER but also the intermediate hops/links across which the packet must be sent. The maximum number of BFER that can be supported in a single BitString or BIFT:SI depends on the number of bits necessary to represent the desired topology between them.

"Desired" topology because it depends on the physical topology, and on the desire of the operator to allow for explicit path steering across every single hop (which requires more bits), or reducing the number of required bits by exploiting optimizations such as unicast (`forward_routed()`), ECMP() or flood (DNC) over "uninteresting" sub-parts of the topology - e.g. parts where different trees do not need to take different paths due to path steering reasons.

The total number of bits to describe the topology vs. the number of BFERs in a BIFT:SI can range widely based on the size of the topology and the amount of alternative paths in it. In a BIER-TE topology crafted by a BIER-TE expert, the higher the percentage of non-BFER bits, the higher the likelihood, that those topology bits are not just BIER-TE overhead without additional benefit, but instead that they will allow to express desirable path steering alternatives.

5.3.3. Assigning BFR-id with BIER-TE

BIER-TE forwarding does not use the BFR-id, nor does it require for the BFIR-id field of the BIER header to be set to a particular value. However, other parts of a BIER-TE deployment may need a BFR-id, specifically multicast flow overlay signaling and multicast flow overlay packet disposition, and in that case BFRs need to also have BFR-ids for BIER-TE SDs.

For example, for BIER overlay signaling, BFIRs need to have a BFR-id, because this BFIR BFR-id is carried in the BFIR-id field of the BIER header to indicate to the overlay signaling on the receiving BFER which BFIR originated the packet.

In BIER, $\text{BFR-id} = \text{SI} * \text{BSL} + \text{BP}$, such that the SI and BP of a BFER can be calculated from the BFR-id and vice versa. This also means that every BFR with a BFR-id has a reserved BP in an SI, even if that is not necessary for BIER forwarding, because the BFR may never be a BFER but only a BFIR.

In BIER-TE, for a non-leaf BFER, there is usually a single BP for that BFER with a `local_decap()` adjacency on the BFER. The BFR-id for such a BFER can therefore be determined using the same procedure as in (non-TE) BIER: $\text{BFR-id} = \text{SI} * \text{BSL} + \text{BP}$.

As explained in Section 5.1.3, leaf BFERs do not need such a unique `local_decap()` adjacency. Likewise, BFIRs that are not also BFERs may not have a unique `local_decap()` adjacency either. For all those BFIRs and (leaf) BFERs, the controller needs to determine unique BFR-ids that do not collide with the BFR-ids derived from the non-leaf BFER `local_decap()` BPs.

While this document defines no requirements on how to allocate such BFR-id, a simple option is to derive it from the (SI,BP) of an adjacency that is unique to the BFR in question. For a BFIR this can be the first adjacency only populated on this BFIR, for a leaf-BFER, this could be the first BP with an adjacency towards that BFER.

5.3.4. Mapping from BFR to BitStrings with BIER-TE

In BIER, applications of the flow overlay on a BFIR can calculate the (SI,BP) of a BFER from the BFR-id of the BFER and can therefore easily determine the BitStrings for a BIER packet to a set of BFERs with known BFR-ids.

In BIER-TE this mapping needs to be equally supported for flow overlays. This section outlines two core options, based on what type of Tree Engineering the BIER-TE controller needs to perform for a particular application.

"Independent branches": For a given flow overlay instance, the branches from a BFIR to every BFER are calculated by the BIER-TE controller to be independent of the branches to any other BFER. Shortest path trees are the most common examples of trees with independent branches.

"Interdependent branches": When a BFER is added or deleted from a particular distribution tree, the BIER-TE controller has to recalculate the branches to other BFER, because they may need to change. Steiner trees are examples of interdependent branch trees.

If "independent branches" are used, the BIER-TE Controller can signal to the BFIR flow overlay for every BFER an SI:BitString that represents the branch to that BFER. The flow overlay on the BFIR can then independently of the controller calculate the SI:BitString for all desired BFERs by OR'ing their BitStrings. This allows for flow overlay applications to operate independently of the controller whenever it needs to determine which subset of BFERs need to receive a particular packet.

If "interdependent branches" are required, the application would need to inquire the SI:BitString for a given set of BFER whenever the set changes.

Note that in either case (unlike in BIER), the bits may need to change upon link/node failure/recovery, network expansion and network resource consumption by other traffic as part of traffic engineering goals (e.g.: re-optimization of lower priority traffic flows). Interactions between such BFIR applications and the BIER-TE Controller do therefore need to support dynamic updates to the SI:BitStrings.

Communications between the BFIR flow overlay and the BIER-TE controller requires some way to identify the BFER. If BFR-ids are used in the deployment, as outlined in Section 5.3.3, then those are the natural BFR identifier. If BFR-ids are not used, then any other unique identifier, such as the BFR-prefix of the BFR ([RFC8279]) could be used.

5.3.5. Assigning BFR-ids for BIER-TE

It is not currently determined if a single sub-domain could or should be allowed to forward both (non-TE) BIER and BIER-TE packets. If this should be supported, there are two options:

- A. BIER and BIER-TE have different BFR-id in the same sub-domain. This allows higher replication efficiency for BIER because their BFR-id can be assigned sequentially, while the BitStrings for BIER-TE will have also the additional bits for the topology. There is no relationship between a BFR BIER BFR-id and its BIER-TE BFR-id.
- B. BIER and BIER-TE share the same BFR-id. The BFR-ids are assigned as explained above for BIER-TE and simply reused for BIER. The replication efficiency for BIER will be as low as that for BIER-TE in this approach.

5.3.6. Example bit allocations

5.3.6.1. With BIER

Consider a network setup with a BSL of 256 for a network topology as shown in Figure 17. The network has 6 areas, each with 170 BFERs, connecting via a core with 4 (core) BFRs. To address all BFERs with BIER, 4 SIs are required. To send a BIER packet to all BFER in the network, 4 copies need to be sent by the BFIR. On the BFIR it does not make a difference how the BFR-ids are allocated to BFER in the network, but for efficiency further down in the network it does make a difference.

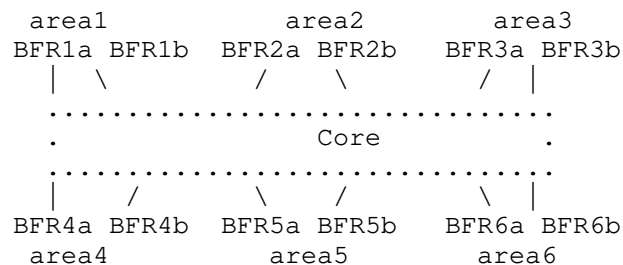


Figure 17: Scaling BIER-TE bits by reuse

With random allocation of BFR-id to BFER, each receiving area would (most likely) have to receive all 4 copies of the BIER packet because there would be BFR-id for each of the 4 SIs in each of the areas. Only further towards each BFER would this duplication subside - when each of the 4 trees runs out of branches.

If BFR-ids are allocated intelligently, then all the BFER in an area would be given BFR-id with as few as possible different SIs. Each area would only have to forward one or two packets instead of 4.

Given how networks can grow over time, replication efficiency in an area will then also go down over time when BFR-ids are only allocated sequentially, network wide. An area that initially only has BFR-id in one SI might end up with many SIs over a longer period of growth. Allocating SIs to areas with initially sufficiently many spare bits for growths can help to alleviate this issue. Or renumber BFERs after network expansion. In this example one may consider to use 6 SIs and assign one to each area.

This example shows that intelligent BFR-id allocation within at least sub-domain 0 can even be helpful or even necessary in BIER.

5.3.6.2. With BIER-TE

In BIER-TE one needs to determine a subset of the physical topology and attached BFERs so that the "desired" representation of this topology and the BFER fit into a single BitString. This process needs to be repeated until the whole topology is covered.

Once bits/SIs are assigned to topology and BFERs, BFR-id is just a derived set of identifiers from the operator/BIER-TE Controller as explained above.

Every time that different sub-topologies have overlap, bits need to be repeated across the BitStrings, increasing the overall amount of bits required across all BitString/SIs. In the worst case, one assigns random subsets of BFERs to different SIs. This will result in an outcome much worse than in (non-TE) BIER: It maximizes the amount of unnecessary topology overlap across SI and therefore reduces the number of BFER that can be reached across each individual SI. Intelligent BFER to SI assignment and selecting specific "desired" subtopologies can minimize this problem.

To set up BIER-TE efficiently for the topology of Figure 17, the following bit allocation method can be used. This method can easily be expanded to other, similarly structured larger topologies.

Each area is allocated one or more SIs depending on the number of future expected BFERs and number of bits required for the topology in the area. In this example, 6 SIs, one per area.

In addition, we use 4 bits in each SI: bia, bib, bea, beb: (b)it (i)ngress (a), (b)it (i)ngress (b), (b)it (e)gress (a), (b)it (e)gress (b). These bits will be used to pass BIER packets from any BFIR via any combination of ingress area a/b BFR and egress area a/b BFR into a specific target area. These bits are then set up with the right forward_routed() adjacencies on the BFIR and area edge BFR:

On all BFIRs in an area $j | j=1...6$, bia in each BIFT:SI is populated with the same forward_routed(BFRja), and bib with forward_routed(BFRjb). On all area edge BFR, bea in BIFT:SI= $k | k=1...6$ is populated with forward_routed(BFRka) and beb in BIFT:SI= k with forward_routed(BFRkb).

For BIER-TE forwarding of a packet to a subset of BFERs across all areas, a BFIR would create at most 6 copies, with SI=1...SI=6. In each packet, the bits indicate bits for topology and BFER in that topology plus the four bits to indicate whether to pass this packet via the ingress area a or b border BFR and the egress area a or b border BFR, therefore allowing path steering for those two "unicast" legs: 1) BFIR to ingress area edge and 2) core to egress area edge. Replication only happens inside the egress areas. For BFER in the same area as in the BFIR, these four bits are not used.

5.3.7. Summary

BIER-TE can, like BIER, support multiple SIs within a sub-domain. This allows to apply the mapping $\text{BFR-id} = \text{SI} * \text{BSL} + \text{BP}$. This allows to re-use the BIER architecture concept of BFR-id and therefore minimize BIER-TE specific functions in possible BIER layer control plane mechanisms with BIER-TE, including flow overlay methods and BIER header fields.

The number of BFIR/BFER possible in a sub-domain is smaller than in BIER because BIER-TE uses additional bits for topology.

Sub-domains (SDs) in BIER-TE can be used like in BIER to create more efficient replication to known subsets of BFERs.

Assigning bits for BFERs intelligently into the right SI is more important in BIER-TE than in BIER because of replication efficiency and overall amount of bits required.

6. Security Considerations

If [RFC8296] is used, BIER-TE shares its security considerations.

BIER-TE shares the security considerations of BIER, [RFC8279], with the following overriding or additional considerations.

BIER-TE forwarding explicitly supports unicast "tunneling" of BIER packets via `forward_routed()` adjacencies. The BIER domain security model is based on a subset of interfaces on a BFR that connect to other BFRs of the same BIER domain. For BIER-TE, this security model equally applies to such unicast "tunneled" BIER packets. This does not only include the need to filter received unicast "tunneled" BIER packets to prohibit injection of such "tunneled" BIER packets from outside the BIER domain, but also prohibiting `forward_routed()` adjacencies to leak BIER packets from the BIER domain. It SHOULD be possible to configure interfaces to be part of a BIER domain solely for sending and receiving of unicast "tunneled" BIER packets even if the interface can not send/receive BIER encapsulated packets.

In BIER, the standardized methods for the routing underlays are IGPs with extensions to distribute BFR-ids and BFR-prefixes. [RFC8401] specifies the extensions for IS-IS and [RFC8444] specifies the extensions for OSPF. Attacking the protocols for the BIER routing underlay or (non-TE) BIER layer control plane, or impairment of any BFR in a domain may lead to successful attacks against the results of the routing protocol, enabling DoS attacks against paths or the addressing (BFR-id, BFR-prefixes) used by BIER.

The reference model for the BIER-TE layer control plane is a BIER-TE controller. When such a controller is used, impairment of an individual BFR in a domain causes no impairment of the BIER-TE control plane on other BFRs. If a routing protocol is used to support `forward_routed()` adjacencies, then this is still an attack vector as in BIER, but only for BIER-TE `forward_routed()` adjacencies, and not other adjacencies.

Whereas IGP routing protocols are most often not well secured through cryptographic authentication and confidentiality, communications between controllers and routers such as those to be considered for the BIER-TE controller/control-plane can be and are much more commonly secured with those security properties, for example by using Secure Shell (SSH), [RFC4253] for NETCONF ([RFC6242]), or via Transport Layer Security (TLS), such as [RFC8253] for PCEP, [RFC5440], or [RFC7589] for NETCONF. BIER-TE controllers SHOULD use security equal to or better than these mechanisms.

When any of these security mechanisms/protocols are used for communications between a BIER-TE controller and BFRs, their security considerations apply to BIER-TE. In addition, the security considerations of PCE, [RFC4655] apply.

The most important attack vector in BIER-TE is misconfiguration, either on the BFR themselves or via the BIER-TE controller. Forwarding entries with DNC could be set up to create persistent loops, in which packets only expire because of TTL. To minimize the impact of such attacks (or more likely unintentional misconfiguration by operators and/or bad BIER-TE controller software), the BIER-TE forwarding rules are defined to be as strict in clearing bits as possible. The clearing of all bits with an adjacency on a BFR prohibits that a looping packet creates additional packet amplification through the misconfigured loop on the packet's second or further times around the loop, because all relevant adjacency bits would have been cleared on the first round through the loop. In result, BIER-TE has the same degree of looping packets as possible with unintentional or malicious loops in the routing underlay with BIER or even with unicast traffic.

Deployments where BIER-TE would likely be beneficial may include operational models where actual configuration changes from the controller are only required during non-production phases of the network's life-cycle, such as in embedded networks or in manufacturing networks during e.g. plant reworking/repairs. In these type of deployments, configuration changes could be locked out when the network is in production state and could only be (re-)enabled through reverting the network/installation into non-production state. Such security designs would not only allow to provide additional layers of protection against configuration attacks, but would foremost protect the active production process from such configuration attacks.

7. IANA Considerations

This document requests no action by IANA.

8. Acknowledgements

The authors would like to thank Greg Shepherd, Ijsbrand Wijnands, Neale Ranns, Dirk Trossen, Sandy Zheng, Lou Berger, Jeffrey Zhang, Carsten Borman and Wolfgang Braun for their reviews and suggestions.

Special thanks to Xuesong Geng for shepherding the document and for IESG review/suggestions by Alvaro Retana (responsible AD/RTG), Benjamin Kaduk (SEC), Tommy Pauly (TSV), Zaheduzzaman Sarker (TSV), Eric Vyncke (INT), Martin Vigoureux (RTG), Robert Wilton (OPS), Eric

Kline (INT), Lars Eggert (GEN), Roman Danyliv (SEC), Ines Robles (RTGDIR), Robert Sparks (Gen-ART), Yingzhen Qu (RTGdir), Martin Duke (TSV).

9. Change log [RFC Editor: Please remove]

draft-ietf-bier-te-arch:

13:

Changed Gregs author association/email.

Fixed Nits in -12 from Ben Kaduk.

Fixed Alvaro's concerns: (1) Removed references to SR in Abstract/Overview (2) removed section 4.5.

12:

AD review Alvaro Retana.

Various textual/editorial nits including adding () to all instances of forwarding adjacency name instances.

3.1 Added new paragraph outlining possible use of BGP as RR in BIER-TE controller as core of multicast flow overlay component of BIER-TE.

3.2 added xref's to relevant sections to the listed control plane points.

4.1 rewrote paragraphs of 4.1 leading up to Figure 4. to eliminate any confusion in how the BIFT work and how it compares to the notions in rfc8279, as well as better linking it to the Pseudocode.

Moved SR section into appendix.

TSV review Martin Duke.

Text/editorial nits.

4.4 improved text describing handling of F-BM.

RTGdir review Yingzhen Qu.

Various text/editorial nits.

Added notion that BitStrings represent loop free tree for packet to abstract and intro.

Various text nit and editorial improvements.

Fixed some BFR-id field -> BFIR-id field mistakes.

Capitalized NETCONF/RESTCONF/YANG, added RFC references.

Improved Figure 16 with explicitly two links into BFR3 and explanatory text.

Gen-ART review Robert Sparks.

Various textual nits, editorial improvements.

3.2 Introduced terms "BIER-TE topology control" and "BIER-TE tree control" for the two functional components of the control plane.

3.2.1 - 3.2 change introduces the open RFC-editor issue of appropriate xrfs (to be resolved by RFC-editor).

3.3 Rewrote last paragraph to better describe loop prevention through clearing of bits in BitString.

4.1 Fixed up text/formula describing mapping between bfr-id, SI:BP and SI,BSL and BP. Fix offset bug.

5.3.6.2 Improved description paragraph explaining overlap of topology for different SI.

5.3.7 Improved first summary paragraph.

7. Rephrased applicability statement of control plane protocol security considerations to BIER-TE security.

RTGDIR review Ines Robles.

Fixed up adjacencies in Example 2 and explanation text to be explicit about which BFR not only passes, but also receives the packet.

7. (security considerations). Added paragraph about forward_routed() and prohibiting BIER packet leaking in/out of domain.

IESG review Roman Danyliv (SEC).

Several textual/sentence nits/editorials.

IESG review Lars Eggert (GEN).

Various good editorial word fixed.

Pointer to non-false-positive bloom filter work that looks like it happened after our IETF discussions documented in this doc, so will not add it to doc, but here is URL for folks interested: <https://ieeexplore.ieee.org/document/8486415>.

Did not change "native" to a different word for inclusivity because of my worry there is no established single replacement word, making reading/searching/understanding more difficult.

IESG review Martin Vigoureux (RTG).

Added back reference to RFC8402. Textual fixes.

IESG review Eric Kline (INT).

2.1 Fixed typo in BFR* explanations.

4.3 Added explanatio about MTU handling.

IESG review Eric Vyncke (INT).

Fixed up initial text to introduce various abbreviations.

2.4 refined wording to "with the `_intent_` to easily build common forwarding planes...".

4.2.3 refined text about entropy in ECMP - now taken text from rfc8279.

IESG review Zaheduzzaman Sarker (TSV).

5.1.7 Refined text explaining documentation of ECMP algorithm.

5.3.6.2. fixed range of areas/SI over which to build the example large network BPs - removed explanation of the large network shown to be only used for sources in area 1 (IPTV), because it was a stale explanation.

IESG review Ben Kaduk (round 2):

4.4 Advanced pseudocode still had one wrong "~". Root cause seems to have been day 0 problem in pseudocode written for -01, "~" was inserted in the wrong one of two code lines. Also enhanced textual description and comments in pseudocode, changed variable name AdjacentBits to PktAdjacentBits to avoid confusion with AdjacentBits[SI].

5.1.3 Rewrote last two paragraphs explaining the sharing of bit positions for lead-BFER hopefully better. Also detailed how it interacts with other optimizations and the type of payload BIER-TE packets may carry.

4.4 (from Carsten Borman) changed spacing in pseudocode to be consistent. Fixed {VRF}, clarified pseudocode object syntax, typos.

11: IESG review Ben Kaduk, summary:

One discuss for bug in pseudocode. turned out to be one cahrcrter typo.

Added (non-TE) prefix in places where BIER by itsels had to be better disambiguated.

enhanced text for hub-and-spoke to indicate we're only talking about hub to spoke traffic.

long list ot language fixes/improvement (nits). Thanks a lot!.

add suggestion to SHOULD use known confidentiality protocols between controller and BFR.

10: AD review Alvaro Retana, summary:

Note: rfcdiff shows more changes than actually exist because text moved around.

Summary:

1. restructuring: merged all controller sections under common controller ops main section, moved unfitting stuff out to other parts of doc. Split Intro section into Overview and Intro. Shortened Abstract, moved text into Overview, added sections overview.
2. enhanced/rewrote: 2.3 Comparison with -> Relationship to BIER-TE

3. enhanced/rewrote: 3.2 BIER-TE controller -> BIER-TE control plane, 3.2.1 BIER-TE controller, for consistency with rfc8279
4. additional subsections for Alvaros asks
5. added to: 3.3 BIER-TE forwarding plane (consistency with rfc8279)
6. Enhanced description of 4.3/encap considerations to better explain how BIER/BIER-TE can run together.

Notation: Markers (a), (b), ... at end of points are references from the review discussion with Alvaro to the changes made.

Details:.

Throughout text: changed term spelling to rfc8279 - bit positions, sub-domain, ... (i).

Reset changed to clear, also DNR changed to DNC (Do Not Clear) (q).

Abstract: Shortened. Removed name explanation note (Tree Engineering), (a).

1. Introduction -> Overview: Moved important explanation paragraph from abstract to Introduction. Fixed text, (a).

Added bullet point list explanation of structure of document (e).

Renamed to Overview because that is now more factually correct.

1.1. Fixed bug in example adding bit p15.(l).

2. (New - Introduction): Moved section 1.1 - 1.3 (examples, comparison with BIER-TE) from Introduction into new "Overview" section. Primarily so that "requirements language" section (at end of Introduction) is not in middle of document after all the Introduction.

2.1 Removed discussion of encap, moved to 4.2.2 (m).

2.2 enhanced paragraph suggesting native/overlay topology types, also suggest type hybrid (n).

2.3 Overhauled comparison text BIER/BIER-TE, structured into common, different, not-required-by-te, integration-bier-bier-te. Changed title to "Relationship" to allow including last point. (f).

2.4 moved Hardware forwarding comparison section into section 2 to allow coalescing of sections into section 5 about the controller operations (hardware forwarding was in the middle of it, wrong place). Shortened/improved third paragraph by pointing to BIFT as deciding element for selection between BIER/BIER-TE. Removed notion of experimentation (this now targets standard) (g).

3. (Components): Aligned component name and descriptions better with RFC8279. Now describe exactly same three layers. BIER layer constituted from BIER-TE control plane and BIER-TE forwarding plane. BIER-TE controller is now simply component of BIER-TE control plane. (b).

3.1. shortened/improved paragraph explaining use of SI:BP instead of also bfr-id as index into BIFT, rewrote paragraph talking about reuse of BPs(o).

3.2. rewrote explanation of BIER-TE control plane in the style of RFC8729 Section 4.2 (BIER layer) with numbered points. Note that RFC8729 mixes control and forwarding plane bullet points (this doc does not). Merged text from old sections 2.2.1 and 2.2.3 into list. (b).

3.2.1. Expanded/improved explanation of BIER-TE Controller (b).

3.2.1.1. Added subsection for topology discovery and creation (d).

3.2.1.2. Added subsection for engineered BitStrings as key novel aspect not found in BIER. (X).

3.3. Added numbered list for components of BIER-TE forwarding plane (completing the comparable text from RFC8729 Section 4.2).

3.4 Alvaro does not mind additional example, fixed bugs.

3.5 Removed notion about using IGP BIER extensions for BIER-TE, such as BIFT address ranges. After -10 making use of BIFT clearer, it now looks to authors as if use of IGP extensions would not be beneficial, as long as we do need to use the BIER-TE controller, e.g. unlike in BIER, a BFR could not learn from the IGP information what traffic to send towards a particular BIFT-ID, but instead that is the core of what the controller needs to provide.

4.2.2 Improved text to explain requirement to identify BIER-TE in the tunnel encap and compress description of use-cases (m).

4.2.3 enhanced ECMP text (p).

4.3. rewrote most of Encapsulation Considerations to better explain to Alvaros question re sharing or not sharing SD via BIER/BIER-TE. Added reference to I-D.ietf-bier-non-mpls-bift-encoding as a very helpful example. (f).

4.3 Renamed title to "...Co-Existence with BIER" as this is what it is about and to help finding it from abstract/intro ("co-exist") (j).

4.4. Moved BIER-TE Forwarding Pseudocode here to coalesce text logically. Changed text to better compare with BIER pseudo forwarding code. Numerical list of how F-BM works for BIER-TE. Removed efficiency comparison with BIER (too difficult to provide sufficient justification, derails from focus of section) (j).

4.6. (Requirements) Restructured: Removed notion of "basic" BIER-TE forwarding, simply referring to it now as "mandatory" BIER-TE forwarding. Cleaned up text to have requirements for different adjacencies in different paragraphs. (c).

5. Created new main section "BIER-TE Controller operational considerations", coalesced old sections 4., 5., 7. into this new main section. No text changes. (k).

5.1.9 Added new separate picture instead of referring to a picture later in text, adjusted text (r).

5.3.2 Changed title to not include word "comparison" to avoid this being accounted against Alvaros concern about scattering comparison (IMHO text already has little comparison, so title was misleading) (h).

co-authors internal review:

4.4 Added xref to Figure 5.

5.2.1 Duplicated ring picture, added visuals for described miswiring (s).

5.2.2 replace "topology" with graph (wrong word).

5.3.3 rewrote explanation of how to map BFR-id to SI:BP and assign them, clarified BFR-id is option. Retitled to better explain scope of section.

5.3.4 Removed considerations in 5.3.4 for sharing BFR-id across BIER/BIER-TE (t), changed title to explain how BFIR/BIER-TE controller interactions need some form of identifying BFR but this does not have to be BFR-id.

7. Added new security considerations (u).

09: Incorporated fixes for feedback from Shepherd (Xuesong Geng).

Added references for Bloom Filters and Rate Controlled Service Disciplines.

1.1 Fixed numbering of example 1 topology explanation. Improved language on second example (less abbreviating to avoid confusion about meaning).

1.2 Improved explanation of BIER-TE topology, fixed terminology of graphs (BIER-TE topology is a directed graph where the edges are the adjacencies).

2.4 Fixed and amended routing underlay explanations: detailed why no need for BFER routing underlay routing protocol extensions, but potential to re-use BIER routing underlay routing protocol extensions for non-BFER related extensions.

3.1 Added explanation for VRF and its use in adjacencies.

08: Incorporated (with hopefully acceptable fixes) for Lou suggested section 2.5, TE considerations.

Fixes are primarily to the point to a) emphasize that BIER-TE does not depend on the routing underlay unless `forward_routed()` adjacencies are used, and b) that the allocation and tracking of resources does not explicitly have to be tied to BPs, because they are just steering labels. Instead, it would ideally come from per-hop resource management that can be maintained only via local accounting in the controller.

07: Further reworking text for Lou.

Renamed BIER-PE to BIER-TE standing for "Tree Engineering" after votes from BIER WG.

Removed section 1.1 (introduced by version 06) because not considered necessary in this doc by Lou (for framework doc).

Added [RFC editor pls. remove] Section to explain name change to future reviewers.

06: Concern by Lou Berger re. BIER-TE as full traffic engineering solution.

Changed title "Traffic Engineering" to "Path Engineering"

Added intro section of relationship BIER-PE to traffic engineering.

Changed "traffic engineering" term in text to "path engineering", where appropriate

Other:

Shortened "BIER-TE Controller Host" to "BIER-TE Controller".
Fixed up all instances of controller to do this.

05: Review Jeffrey Zhang.

Part 2:

4.3 added note about leaf-BFER being also a property of routing setup.

4.7 Added missing details from example to avoid confusion with routed adjacencies, also compressed explanatory text and better justification why seed is explicitly configured by controller.

4.9 added section discussing generic reuse of BP methods.

4.10 added section summarizing BP optimizations of section 4.

6. Rewrote/compressed explanation of comparison BIER/BIER-TE forwarding difference. Explained benefit of BIER-TE per-BP forwarding being independent of forwarding for other BPs.

Part 1:

Explicitly use forwarded_connected adjacency in ECMP adjacency examples to avoid confusion.

4.3 Add picture as example for leaf vs. non-leaf BFR in topology. Improved description.

4.5 Example for traffic that can be broadcast -> for single BP in hub&spoke.

4.8.1 Simplified example picture for routed adjacency, explanatory text.

Review from Dirk Trossen:

Fixed up explanation of ICC paper vs. bloom filter.

04: spell check run.

Added remaining fixes for Sandys (Zhang Zheng) review:

4.7 Enhance ECMP explanations:

example ECMP algorithm, highlight that doc does not standardize ECMP algorithm.

Review from Dirk Trossen:

1. Added mentioning of prior work for traffic engineered paths with bloom filters.

2. Changed title from layers to components and added "BIER-TE control plane" to "BIER-TE Controller" to make it clearer, what it does.

2.2.3. Added reference to I-D.ietf-bier-multicast-http-response as an example solution.

2.3. clarified sentence about resetting BPs before sending copies (also forgot to mention DNR here).

3.4. Added text saying this section will be removed unless IESG review finds enough redeeming value in this example given how -03 introduced section 1.1 with basic examples.

7.2. Removed explicit numbers 20%/80% for number of topology bits in BIER-TE, replaced with more vague (high/low) description, because we do not have good reference material Added text saying this section will be removed unless IESG review finds enough redeeming value in this example given how -03 introduced section 1.1 with basic examples.

many typos fixed. Thanks a lot.

03: Last call textual changes by authors to improve readability:

removed Wolfgang Braun as co-authors (as requested).

Improved abstract to be more explanatory. Removed mentioning of FRR (not concluded on so far).

Added new text into Introduction section because the text was too difficult to jump into (too many forward pointers). This primarily consists of examples and the early introduction of the BIER-TE Topology concept enabled by these examples.

Amended comparison to SR.

Changed syntax from [VRF] to {VRF} to indicate its optional and to make idnits happy.

Split references into normative / informative, added references.

02: Refresh after IETF104 discussion: changed intended status back to standard. Reasoning:

Tighter review of standards document == ensures arch will be better prepared for possible adoption by other WGs (e.g. DetNet) or std. bodies.

Requirement against the degree of existing implementations is self defined by the WG. BIER WG seems to think it is not necessary to apply multiple interoperating implementations against an architecture level document at this time to make it qualify to go to standards track. Also, the levels of support introduced in -01 rev. should allow all BIER forwarding engines to also be able to support the base level BIER-TE forwarding.

01: Added note comparing BIER and SR to also hopefully clarify BIER-TE vs. BIER comparison re. SR.

- added requirements section mandating only most basic BIER-TE forwarding features as MUST.

- reworked comparison with BIER forwarding section to only summarize and point to pseudocode section.

- reworked pseudocode section to have one pseudocode that mirrors the BIER forwarding pseudocode to make comparison easier and a second pseudocode that shows the complete set of BIER-TE forwarding options and simplification/optimization possible vs. BIER forwarding. Removed MyBitsOfInterest (was pure optimization).

- Added captions to pictures.

- Part of review feedback from Sandy (Zhang Zheng) integrated.

00: Changed target state to experimental (WG conclusion), updated references, mod auth association.

- Source now on <https://www.github.com/toerless/bier-te-arch>

- Please open issues on the github for change/improvement requests to the document - in addition to posting them on the list (bier@ietf.). Thanks!.

draft-eckert-bier-te-arch:

06: Added overview of forwarding differences between BIER, BIER-TE.

05: Author affiliation change only.

04: Added comparison to Live-Live and BFIR to FRR section (Eckert).

04: Removed FRR content into the new FRR draft [I-D.eckert-bier-te-frr] (Braun).

- Linked FRR information to new draft in Overview/Introduction

- Removed BTAFT/FRR from "Changes in the network topology"

- Linked new draft in "Link/Node Failures and Recovery"

- Removed FRR from "The BIER-TE Forwarding Layer"
- Moved FRR section to new draft
- Moved FRR parts of Pseudocode into new draft
- Left only non FRR parts
- removed `FrrUpDown(..)` and `//FRR` operations in `ForwardBierTePacket(..)`
- New draft contains `FrrUpDown(..)` and `ForwardBierTePacket(Packet)` from `bier-arch-03`
- Moved "BIER-TE and existing FRR to new draft
- Moved "BIER-TE and Segment Routing" section one level up
- Thus, removed "Further considerations" that only contained this section
- Added Changes for version 04

03: Updated the FRR section. Added examples for FRR key concepts. Added BIER-in-BIER tunneling as option for tunnels in backup paths. BIFT structure is expanded and contains an additional match field to support full node protection with BIER-TE FRR.

03: Updated FRR section. Explanation how BIER-in-BIER encapsulation provides P2MP protection for node failures even though the routing underlay does not provide P2MP.

02: Changed the definition of BIFT to be more inline with BIER. In revs. up to -01, the idea was that a BIFT has only entries for a single BitString, and every SI and sub-domain would be a separate BIFT. In BIER, each BIFT covers all SI. This is now also how we define it in BIER-TE.

02: Added Section 5.3 to explain the use of SI, sub-domains and BFR-id in BIER-TE and to give an example how to efficiently assign bits for a large topology requiring multiple SI.

02: Added further detailed for rings - how to support input from all ring nodes.

01: Fixed BFIR -> BFER for section 4.3.

01: Added explanation of SI, difference to BIER ECMP, consideration for Segment Routing, unicast FRR, considerations for encapsulation, explanations of BIER-TE Controller and CLI.

00: Initial version.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.
- [RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <<https://www.rfc-editor.org/info/rfc8296>>.

10.2. Informative References

- [Bloom70] Bloom, B. H., "Space/time trade-offs in hash coding with allowable errors", Comm. ACM 13(7):422-6, July 1970, <<https://dl.acm.org/doi/10.1145/362686.362692>>.
- [I-D.eckert-bier-te-frr] Eckert, T., Cauchie, G., Braun, W., and M. Menth, "Protection Methods for BIER-TE", Work in Progress, Internet-Draft, draft-eckert-bier-te-frr-03, 5 March 2018, <<https://www.ietf.org/archive/id/draft-eckert-bier-te-frr-03.txt>>.
- [I-D.ietf-bier-multicast-http-response] Trossen, D., Rahman, A., Wang, C., and T. Eckert, "Applicability of BIER Multicast Overlay for Adaptive Streaming Services", Work in Progress, Internet-Draft,

draft-ietf-bier-multicast-http-response-06, 10 July 2021, <<https://www.ietf.org/archive/id/draft-ietf-bier-multicast-http-response-06.txt>>.

- [I-D.ietf-bier-non-mpls-bift-encoding]
Wijnands, I., Mishra, M., Xu, X., and H. Bidgoli, "An Optional Encoding of the BIFT-id Field in the non-MPLS BIER Encapsulation", Work in Progress, Internet-Draft, draft-ietf-bier-non-mpls-bift-encoding-04, 30 May 2021, <<https://www.ietf.org/archive/id/draft-ietf-bier-non-mpls-bift-encoding-04.txt>>.
- [I-D.ietf-bier-te-yang]
Zhang, Z., Wang, C., Chen, R., Hu, F., Sivakumar, M., and H. Chen, "A YANG data model for Tree Engineering for Bit Index Explicit Replication (BIER-TE)", Work in Progress, Internet-Draft, draft-ietf-bier-te-yang-04, 7 November 2021, <<https://www.ietf.org/archive/id/draft-ietf-bier-te-yang-04.txt>>.
- [I-D.ietf-roll-ccast]
Bergmann, O., Bormann, C., Gerdes, S., and H. Chen, "Constrained-Cast: Source-Routed Multicast for RPL", Work in Progress, Internet-Draft, draft-ietf-roll-ccast-01, 30 October 2017, <<https://www.ietf.org/archive/id/draft-ietf-roll-ccast-01.txt>>.
- [I-D.ietf-teas-rfc3272bis]
Farrel, A., "Overview and Principles of Internet Traffic Engineering", Work in Progress, Internet-Draft, draft-ietf-teas-rfc3272bis-16, 24 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-teas-rfc3272bis-16.txt>>.
- [ICC]
Reed, M. J., Al-Naday, M., Thomos, N., Trossen, D., Petropoulos, G., and S. Spirou, "Stateless multicast switching in software defined networks", IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 2016, May 2016, <<https://ieeexplore.ieee.org/document/7511036>>.
- [RCSD94]
Zhang, H. and D. Domenico, "Rate-Controlled Service Disciplines", Journal of High-Speed Networks, 1994, May 1994, <<https://dl.acm.org/doi/10.5555/2692227.2692232>>.
- [RFC4253]
Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <<https://www.rfc-editor.org/info/rfc4253>>.

- [RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <<https://www.rfc-editor.org/info/rfc4456>>.
- [RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <<https://www.rfc-editor.org/info/rfc4655>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <<https://www.rfc-editor.org/info/rfc5440>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <<https://www.rfc-editor.org/info/rfc7589>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<https://www.rfc-editor.org/info/rfc7752>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC7988] Rosen, E., Ed., Subramanian, K., and Z. Zhang, "Ingress Replication Tunnels in Multicast VPN", RFC 7988, DOI 10.17487/RFC7988, October 2016, <<https://www.rfc-editor.org/info/rfc7988>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.

- [RFC8253] Lopez, D., Gonzalez de Dios, O., Wu, Q., and D. Dhody, "PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)", RFC 8253, DOI 10.17487/RFC8253, October 2017, <<https://www.rfc-editor.org/info/rfc8253>>.
- [RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <<https://www.rfc-editor.org/info/rfc8345>>.
- [RFC8401] Ginsberg, L., Ed., Przygienda, T., Aldrin, S., and Z. Zhang, "Bit Index Explicit Replication (BIER) Support via IS-IS", RFC 8401, DOI 10.17487/RFC8401, June 2018, <<https://www.rfc-editor.org/info/rfc8401>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8444] Psenak, P., Ed., Kumar, N., Wijnands, IJ., Dolganow, A., Przygienda, T., Zhang, J., and S. Aldrin, "OSPFv2 Extensions for Bit Index Explicit Replication (BIER)", RFC 8444, DOI 10.17487/RFC8444, November 2018, <<https://www.rfc-editor.org/info/rfc8444>>.
- [RFC8556] Rosen, E., Ed., Sivakumar, M., Przygienda, T., Aldrin, S., and A. Dolganow, "Multicast VPN Using Bit Index Explicit Replication (BIER)", RFC 8556, DOI 10.17487/RFC8556, April 2019, <<https://www.rfc-editor.org/info/rfc8556>>.

Appendix A. BIER-TE and Segment Routing (SR)

SR ([RFC8402]) aims to enable lightweight path steering via loose source routing. Compared to its more heavy-weight predecessor RSVP-TE, SR does for example not require per-path signaling to each of these hops.

BIER-TE supports the same design philosophy for multicast. Like in SR, it relies on source-routing - via the definition of a BitString. Like SR, it only requires to consider the "hops" on which either replication has to happen, or across which the traffic should be steered (even without replication). Any other hops can be skipped via the use of routed adjacencies.

BIER-TE bit position (BP) can be understood as the BIER-TE equivalent of "forwarding segments" in SR, but they have a different scope than SR forwarding segments. Whereas forwarding segments in SR are global or local, BPs in BIER-TE have a scope that is the group of BFR(s) that have adjacencies for this BP in their BIFT. This can be called "adjacency" scoped forwarding segments.

Adjacency scope could be global, but then every BFR would need an adjacency for this BP, for example a `forward_routed()` adjacency with encapsulation to the global SR SID of the destination. Such a BP would always result in ingress replication though (as in [RFC7988]). The first BFR encountering this BP would directly replicate to it. Only by using non-global adjacency scope for BPs can traffic be steered and replicated on non-ingress BFR.

SR can naturally be combined with BIER-TE and help to optimize it. For example, instead of defining bit positions for non-replicating hops, it is equally possible to use segment routing encapsulations (e.g. SR-MPLS label stacks) for the encapsulation of "forward_routed" adjacencies.

Note that (non-TE) BIER itself can also be seen to be similar to SR. BIER BPs act as global destination Node-SIDs and the BIER BitString is simply a highly optimized mechanism to indicate multiple such SIDs and let the network take care of effectively replicating the packet hop-by-hop to each destination Node-SID. What BIER does not allow is to indicate intermediate hops, or in terms of SR the ability to indicate a sequence of SID to reach the destination. This is what BIER-TE and its adjacency scoped BP enables.

Authors' Addresses

Toerless Eckert (editor)
Futurewei Technologies Inc.
2330 Central Expy
Santa Clara, 95050
United States of America
Email: tte+ietf@cs.fau.de

Michael Menth
University of Tuebingen
Email: menth@uni-tuebingen.de

Gregory Cauchie
KOEVOO
Email: gregory@koevoo.tech

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 2, 2018

D. Purkayastha
A. Rahman
D. Trossen
InterDigital Communications, LLC
March 1, 2018

Multicast HTTP using BIER
draft-purkayastha-bier-multicast-http-00

Abstract

HTTP Level multicast, using BIER, is described in the working group use case document. Specifically, it describes how individual HTTP responses can utilize a single BIER multicast response, utilizing an edge-based service routing components on top of the BIER transport. In order to enable the use case, the document describes additional functions in the ingress and egress nodes to the BIER network. These functions are assumed to be part of the BIER multicast overlay.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 2, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions used in this document	2
3. Background	3
3.1. Applicability	3
3.2. State Of The Art	4
4. Requirements	6
5. HTTP Multicast Overlay Components	6
6. HTTP Multicast Overlay Operations	7
7. Required Protocol Changes	9
8. Next Steps	9
9. IANA Considerations	9
10. Security Considerations	9
11. Informative References	9
Authors' Addresses	10

1. Introduction

BIER Use Cases document [I-D.ietf-bier-use-cases] describes an "HTTP Level Multicast" scenario, where HTTP Responses are carried over a BIER multicast infrastructure to multiple clients. HTTP-level clients benefit from the dynamic multicast group formation enabled by BIER. For this, the server side Service Router (SR), creates a list of outstanding client side Service Router (SR) requests for the same HTTP resource. When a response is available, BIER forwarding information is retrieved and used to send the HTTP response.

In this draft, we introduce the requirements for a BIER multicast overlay realizing this use case. It also describes the necessary functions that form the BIER multicast overlay and the operations that enable the desired "HTTP Level Multicast" behavior. We describe a list of protocols needed for the realization of the individual operations.

We conclude with future steps and seek input from the WG.

2. Conventions used in this document

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 [RFC2119].

3. Background

3.1. Applicability

With the extensive use of "web technology", "distributed services" and availability of heterogeneous network, HTTP has effectively transitioned into the common transport for E2E communication across the web. HTTP request and response is used in media streaming and delivery applications. In such scenarios, where semi-synchronous access to the same resource occurs (such as watching prominent videos over Netflix or similar platforms or liveTV over HTTP), traffic grows linearly with the number of viewers since the HTTP-based server will provide an HTTP response to each individual viewer. This poses a significant burden on operators in terms of costs and on users in terms of likely degradation of quality. BIER can greatly reduce this burden, as described in the use case [I-D.ietf-bier-use-cases], by utilizing the BIER routing overlay to transport a single HTTP response to several edge nodes. Edge nodes may have additional logic to 'route' the HTTP-based service from and to the individual clients. The path-based routing applied in BIER is particularly appealing since it will allow for building those multicast relations per HTTP request/response relation in an ad-hoc manner, thereby improving flexibility and utilization even further.

Applicability of "HTTP Level Multicast" is not only restricted for Video streaming and delivery. It may be applied in other use cases such as Virtual Reality, V2X where users may access, in semi-synchronous way, the same resource.

Consider a virtual reality use case where several users are joining a VR session at the same time, e.g., centered around a joint event. Hence, due to the temporal correlation of the VR sessions, we can assume that multiple requests are sent for the same content at any point, particularly when viewing angles of VR clients are similar or the same. The "HTTP Level Multicast" use case allows reducing the load on the network and faster delivery of content.

In a V2X scenario, at a particular location, as many vehicles enters, they may request geo-location, safety related information from the same content server. The requests may be semi-synchronous or close in chronological order. "HTTP Level Multicast" will reduce the flood of HTTP response and latency in delivering the information to the vehicles.

As part of POINT/RIFE EU Horizon 2020 project, HTTP Level Multicast use case has been executed on SDN based and ICN based underlay network, as described in the [I-D.irtf-icnrg-deployment-guidelines]. "HTTP multicast" demonstrated benefits in HTTP-level streaming video

delivery, when deployed on POINT test bed with 80+ nodes. This draft [I-D.irtf-icnrg-deployment-guidelines] also describes protocol requirements to enable HTTP multicast to work on ICN underlay.

This use case completely works as an overlay on BIER. The multicast here is ad-hoc, i.e., the multicast relations are built at the level of each HTTP response and can therefore vary from one request/response transaction to others. Returning to our VR scenario above, the multicast relations are being formed for each request for a VR video chunk. If more than one VR client has requested said chunk at the time defined by the response delay for delivering the chunk from the video server, BIER multicast relations are formed in an ad-hoc manner and the response is sent to the clients with outstanding requests to the same chunk via a BIER-level multicast. Note that these multicast relations are highly dynamic. For instance, in the case of the VR scenario, changes in viewing angles by VR clients will result in completely access patterns to chunks at the next retrieval. This differs from edge multicast flow aggregators which assume stable multicast relations that can be mapped onto, e.g., IP multicast.

3.2. State Of The Art

This use case describes how a single HTTP Response, which represents N number of responses for the same resource, can be directed towards correct ingress point in a fast and dynamic way. The routing decision is abstracted at HTTP level, instead of the traditional approach where routing decision for a service request is made at Layer 3 after resolution of the service name onto a locator such as an IP address. This means HTTP requests and responses are routed based on the URI associated with the request. URI is simply put, name of a resource. Using URI to identify Source and Destination, HTTP requests are routed using a path-based forwarding. To summarize, routing of HTTP request/response can be done based on named services and HTTP is used as a special named (application layer) service. The routing of those request is done via a service router (as shown in Fig. 1), which utilizes a path-based approach to forwarding.

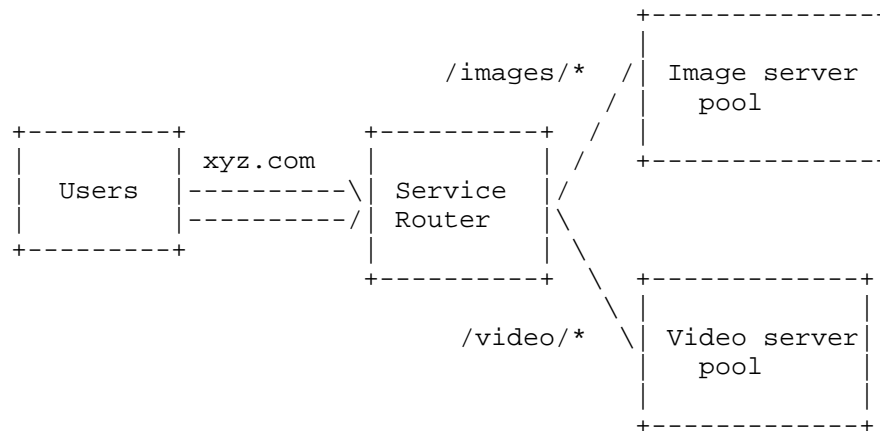


Figure 1: Path Based Routing

This service router is configured with rules to forward requests based on the URL path. E.g in case of multiple micro-services being run, traffic can be routed to multiple back-end services using path-based routing. For example, general requests are routed to one target group and requests to render images to another target group.

"HTTP Level multicast" may work on existing transport technology using SDN based forwarding [Reed2016]. This option utilizes path-based forwarding through SDN-based wildcard matching fields, supported with OF1.2+ [Reed2016]. It can be embedded into slicing approach of underlying transport infrastructure by leaving typical slicing fields available (e.g., VLAN tags). The forwarding utilizes the Ethernet frame format at Layer 2, representing the topological links of a specific forwarding path in the transport network as unique bits in a fixed size bit array. For the latter, the approach utilizes the IPv6 source and destination fields for storing the bit array information (in a simple version for this forwarding, this limits the topology to 256 links but extensions schemes are possible, which are left out of this document at this stage). As mentioned, the SDN forwarding action is a simple wildcard matching, supported with OF1.2+, with the wildcard representing the unique bit of a switch-specific output port. With that, the switch needs to consider as many forwarding rules as switch local output ports, see [Reed2016] for more information.

4. Requirements

A realization for the "HTTP multicast" use case may have the following requirements:

- o MUST support multiple FQDN-based service endpoints to exist in the overlay
- o MUST send FQDN-based service requests at the network level to a suitable FQDN-based service endpoint via policy-based selection of appropriate path information
- o MUST allow for multicast delivery of HTTP response to same HTTP request URI
- o MUST provide direct path mobility, where the path between the egress and ingress Service Routers(SR) can be determined as being optimal (e.g., shortest path or direct path to a selected instance), is needed to avoid the use of anchor points and further reduce service-level latency

5. HTTP Multicast Overlay Components

Let us formulate the architecture of the BIER multicast overlay for the scenario outlined in [I-D.ietf-bier-use-cases]. This overlay is shown in Figure 2 below.

The multicast overlay is formed by the BFIR and BFER of the BIER layer and the additional SR and PCE elements shown in the figure. When connecting to a standard IP routed peering network, a special SR is utilized, shown as the border GW in the figure.

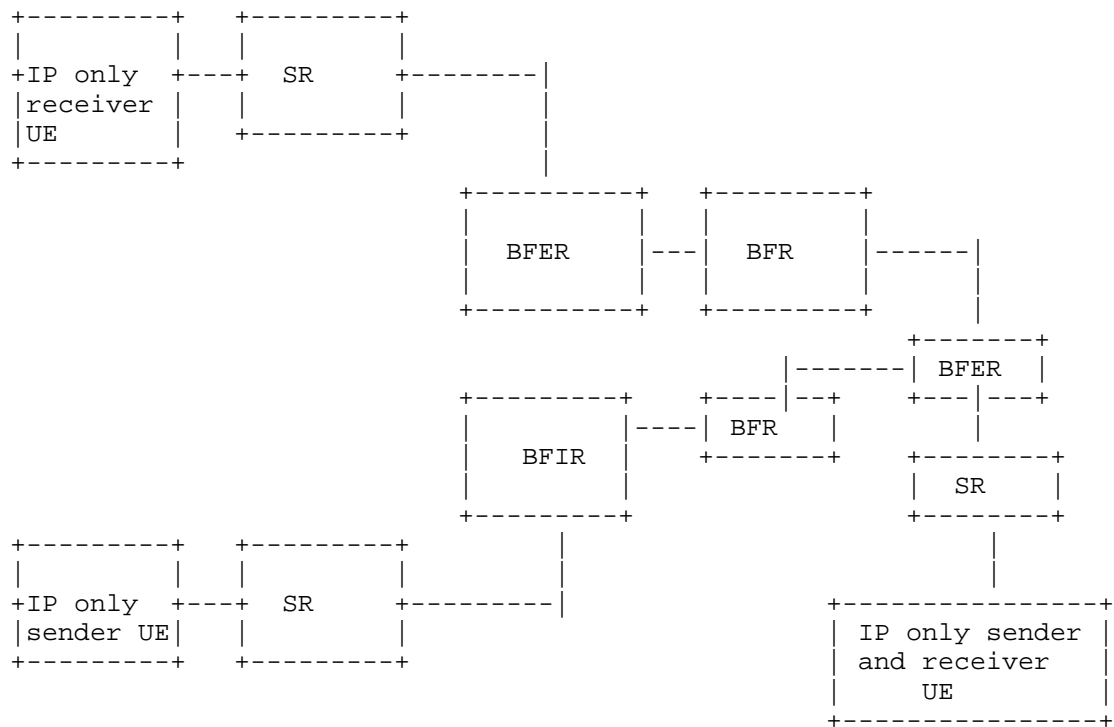


Figure 2: BIER Multicast Overlay for HTTP Multicast Use case

6. HTTP Multicast Overlay Operations

As shown in Figure 2, the multicast overlay includes a function called PCE (Path Computation Element function), which is responsible for selecting the correct multicast end point and possibly realizing path policy enforcement. The result of the selection is a BIER path identifier, which is delivered to the SR upon initial path computation request (i.e., when sending a request to or response for a specific URL for the first time). The path identifier is utilized for any future request for a given URL-based request. All service end points indicate availability to the PCE through a registration procedure, the PCE will instruct all SRs to invalidate previous path identifiers to the specific URL. This may result in an initial path computation request at the next service request forwarding. Through this, the newly registered service endpoint might be utilized if the policy-governed path computation selects said service instance.

In the architecture of Figure 2, an HTTP request is sent by an IP-based device towards the FQDN of the server defined in the HTTP request.

At the client facing SR, the HTTP request is terminated at the HTTP level at a local HTTP proxy. We assume termination on the client side at Layer 3 and above protocols, such as TCP. Server side SR at the egress, terminates any transport protocol on the outgoing (server) side. These terminating functions are assumed to be part of the client/server SR.

If no local BIER forwarding information exists to the server SR, a path computation entity (PCE) is consulted, which calculates a unicast path from the BFIR to which the client SR is connected to the BFER to which the server SR is connected. The PCE provides the forwarding information to the client SR, which in turn caches the result.

Ultimately, the HTTP request is forwarded by the client SR towards the server-facing SR via the local BFIR. We assume a (TCP-friendly) transport protocol being used for the transmission between client and server SR while not mandating the use of TCP for this transmission.

Upon arrival of an HTTP request at the server SR, the server SR proxy forwards the HTTP request as a well-formed HTTP request locally to the server.

If no BIER forwarding information exists for the reverse direction towards the requesting client SR, this information is requested from the PCE, similar to the operation in forward direction.

Upon arrival of any further client SR request at the server SR to an HTTP request whose response is still outstanding, the client SR is added to an internal request table. Optionally, the request is suppressed from being sent to the server.

Upon arrival of an HTTP response at the server SR, the server SR consults its internal request table for any outstanding HTTP requests to the same request. The server SR retrieves the stored BIER forwarding information for the reverse direction for all outstanding HTTP requests and determines the path information to all client SRs through a binary OR over all BIER forwarding identifiers with the same SI field. This newly formed joint BIER multicast response identifier is used to send the HTTP response across the network.

7. Required Protocol Changes

For the operations outlined in the previous section, we foresee the following protocol changes may be required:

- o SR-to-SR protocol for HTTP: Map HTTP to BIER message exchange between client and server SRs
- o SR-PCE protocol: Used for path computation and delivery of BIER routing information as well as path updates
- o Registration protocol: Used to register FQDN service endpoints

8. Next Steps

Given the importance of HTTP-based services, we therefore suggest to include an additional Applicability Statement documenting how BIER can be applied to aggregate HTTP responses over a BIER infrastructure (which we term as "HTTP Multicast"). This new proposed Applicability Statement document will describe how BIER can be applied to implement efficient, dynamic multicast support for the delivery of HTTP responses to individual HTTP requests for the same resource.

9. IANA Considerations

This document requests no IANA actions.

10. Security Considerations

TBD.

11. Informative References

[I-D.ietf-bier-use-cases]

Kumar, N., Asati, R., Chen, M., Xu, X., Dolganow, A., Przygienda, T., Gulko, A., Robinson, D., Arya, V., and C. Bestler, "BIER Use Cases", draft-ietf-bier-use-cases-06 (work in progress), January 2018.

[I-D.irtf-icnrg-deployment-guidelines]

Rahman, A., Trossen, D., Kutscher, D., and R. Ravindran, "Deployment Considerations for Information-Centric Networking (ICN)", draft-irtf-icnrg-deployment-guidelines-00 (work in progress), February 2018.

- [Reed2016] Reed, M., Al-Naday, M., Thomas, N., Trossen, D., Petropoulos, G., and S. Spirou, "Stateless multicast switching in software defined networks", ICC 2016, 2016.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

Authors' Addresses

Debashish Purkayastha
InterDigital Communications, LLC
Conshohocken
USA

Email: Debashish.Purkayastha@InterDigital.com

Akbar Rahman
InterDigital Communications, LLC
Montreal
Canada

Email: Akbar.Rahman@InterDigital.com

Dirk Trossen
InterDigital Communications, LLC
64 Great Eastern Street, 1st Floor
London EC2A 3QR
United Kingdom

Email: Dirk.Trossen@InterDigital.com
URI: <http://www.InterDigital.com/>

BIER
Internet-Draft
Intended status: Standards Track
Expires: September 4, 2018

P. Thubert, Ed.
Cisco
T. Eckert
Huawei
Z. Brodard
Ecole Polytechnique
H. Jiang
Telecom Bretagne
March 3, 2018

BIER-TE extensions for Packet Replication and Elimination Function
(PREF) and OAM
draft-thubert-bier-replication-elimination-03

Abstract

This specification extends Bit Index Explicit Replication - Traffic Engineering (BIER-TE) forwarding to support in the data plane the DetNet Packet Replication and Elimination Functions (PREF). It also provides traceability of links/adjacencies where replication and loss happen, in a manner that is agnostic to the forwarding information (OAM).

Status of This Memo

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

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

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

This Internet-Draft will expire on September 4, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. On BIER - Traffic Engineering	3
4. BIER-TE-based Replication and Elimination Control	4
5. Elimination Function (Normative)	9
6. Summary	11
7. Implementation Status	12
8. Security considerations	12
9. IANA Considerations	12
10. Acknowledgements	12
11. References	13
11.1. Normative References	13
11.2. Informative References	13
Authors' Addresses	14

1. Introduction

Deterministic Networking (DetNet) [I-D.ietf-detnet-problem-statement] provides a capability to carry unicast or multicast data flows for real-time applications with extremely low data loss rates and known upper bound maximum latency [I-D.ietf-detnet-architecture].

DetNet applies to multiple environments where there is a desire to replace a point to point serial cable or a multidrop bus by a switched or routed infrastructure, in order to scale, lower costs, and simplify management. One classical use case is found in particular in the context of the convergence of IT with Operational Technology (OT), also referred to as the Industrial Internet. But there are many others use cases [I-D.ietf-detnet-use-cases], for instance in professional audio and video, automotive, radio fronthauls, etc..

The DetNet data plane alternatives [I-D.ietf-detnet-dp-alt] studies the applicability of existing and emerging dataplane techniques that can be leveraged to enable DetNet properties in IP networks. One critical feature in the dataplane is traceability, the capability to control the activity of intermediate nodes on a packet. For instance, if Replication and Elimination is applied to a packet, then it is desirable to determine which node performed a certain copy of

that packet that is circulating in the network. Likewise, engineered paths are required to support redundant transmission across disjoint paths in support of DetNets PREF functions.

Traceability belongs to Operations, Administration, and Maintenance (OAM) which is the toolset for fault detection and isolation, and for performance measurement. More can be found on OAM Tools in "An Overview of Operations, Administration and Maintenance (OAM) Tools" [I-D.ietf-opsawg-oam-overview].

This document proposes a new set to mechanisms based on [RFC8279] (BIER) and more specifically BIER Traffic Engineering [I-D.ietf-bier-te-arch] (BIER-TE) to control the process or Packet Replication and Elimination Functions (PREF), and provide traceability of these operations, in the DetNet dataplane. An adjacency, which is represented by a bit in the BIER header, can correspond in the dataplane to an Ethernet hop, a Label Switched Path, or it can correspond to an IPv6 loose or strict source routed path.

BIER-TE was primarily designed to carry multicast traffic, but there is nothing prohibiting for it to be used with unicast traffic, and the authors of this document think that for networks whose size requirement match the supportable bitstring length (BSL) in BIER, it can be a good choice as the forwarding plane specifically for DetNet type traffic for both multicast and unicast traffic because it would be a common solution for unicast and multicast (limiting the number of different technologies a DetNet solution requires) and likely provides the most flexible support for path engineering, replication and elimination (PREF) and the novel OAM method described in this document.

2. Terminology

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 [RFC2119].

3. On BIER - Traffic Engineering

[RFC8279] (BIER) is a network plane replication technique that was initially intended as a new method for multicast distribution. In a nutshell, a BIER header includes a bitmap that explicitly signals the listeners that are intended for a particular packet, which means that 1) the sender is aware of the individual listeners and 2) the BIER control plane is a simple extension of the unicast routing as opposed to a dedicated multicast data plane, which represents a considerable

reduction in OPEX. For this reason, the technology faces a lot of traction from Service Providers.

The simplicity of the BIER technology makes it very versatile as a network plane signaling protocol. Already, a new Traffic Engineering variation is emerging that uses bits to signal segments along a TE path.

While BIER-TE was like BIER primarily developed for multicast traffic, the authors think that it can equally be attractive for unicast traffic requiring the DetNet resilience of multiple transitions. If the topology of the network can well be represented by standard BIER-TE bitstring sizes of e.g.: up to 256 bits, then this would allow for a single technology for both unicast and multicast.

BIER-TE supports a Traffic Engineered forwarding plane by explicit hop-by-hop forwarding and loose hop forwarding of packets.

From the BIER-TE architecture, the key differences over BIER are:

- o BIER-TE replaces in-network autonomous path calculation by explicit paths calculated off path for example by a BIER-TE controller host.
- o In BIER-TE every BitPosition of the BitString of a BIER-TE packet indicates one or more adjacencies - instead of a BFER as in BIER. processing packets as a destination (BFER) is one of the possible adjacency types.
- o BIER-TE in each BFR has no routing table but only a BIER-TE Forwarding Table (BIFT) indexed by SI:BitPosition and populated with only those adjacencies to which the BFR should replicate packets to.

The generic view of an adjacency can be over a link, a tunnel or along a path segment.

4. BIER-TE-based Replication and Elimination Control

This document only needs to introduce new functionality to support the Elimination Function and OAM. Creation of appropriate BIER-TE packets is subject to to existing work.

In the solution described below, the encapsulation/insertion of flow-identification and sequence number into packets is performed by a function on the BFIR outside the scope of this document. A companion document draft-huang-bier-te-encapsulation defines an encapsulation for BIER-TE and BIER that can support flow-id and sequence-number ID. Other encapsulations can be used as well, as long as they provide

these signaling elements and are supported by the Elimination Function described in this document (e.g.: that the EF can read these fields and therefore remove duplicates). In the remainder of this document we will call this the extended BIER encapsulation and assume that it is used when describing examples. Unless otherwise noted, we assume that the BFIR performs encapsulation of some data flow packets with an extended BIER header, indicates BIER-TE forwarding in it and fills in flow-id and sequence number. It then fills in the bitstring with two (or more) alternative paths/DAGs and sends off the packets into the BIER-TE domain, replicating it itself if so indicated by the bitstring.

In a nutshell, BIER-TE is used as follows:

- o A controller computes a complex path, sometimes called a track, which takes the general form of a ladder. The steps and the side rails between them are the adjacencies that can be activated on demand on a per-packet basis using bits in the BIER header.

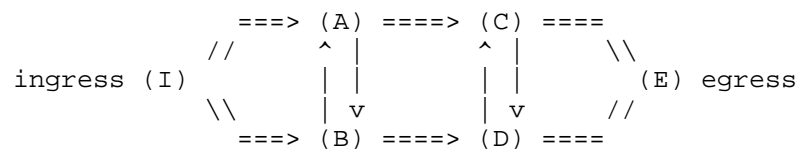


Figure 1: Ladder Shape with Replication and Elimination Points

- o The controller assigns a BIER domain, and inside that domain, assigns bits to the adjacencies. The controller assigns each bit to a replication node that sends towards the adjacency, for instance the ingress router into a segment that will insert a routing header in the packet. A single bit may be used for a step in the ladder, indicating the other end of the step in both directions.

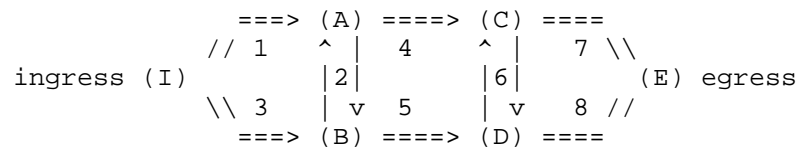


Figure 2: Assigning Bits

- o The controller activates the replication by deciding the setting of the bits associated with the adjacencies. This decision can be modified at any time, but takes the latency of a controller round trip to effectively take place. Below is an example that uses Replication and Elimination to protect the A->C adjacency. The "(EF)" in the following pictures Owner column indicates the fact that that BFR will perform the "Elimination Function" for received BIER-TE packets before further processing/copying them. In this example, only C performs EF. A (1) in the Example Bitstring indicates that the bit is set, but that the actual adjacency is not used by packets because this bit is shared with another adjacency and the overall bitstring will make the packet only use that other adjacency. This applies to bits 2 and 6.

Bit #	Adjacency	Owner	Example Bitstring
1	I->A	I	1
2	A->B	A	1
	B->A	B	(1)
3	I->B	I	0
4	A->C	A	1
5	B->D	B	1
6	C->D	C (EF)	(1)
	D->C	D	
7	C->E	C (EF)	1
8	D->E	D	0

Replication and Elimination Protecting A->C

Table 1: Controlling Replication

- o The BIER header with the controlling BitString, flow-id and sequence number is injected in the packet by the ingress node I (BFIR). That node may act as a replication point, in which case it may issue multiple copies of the packet, but for the purpose of this example it will not do it, so that the two paths used in this example only go from A to C, and therefore require explicit path engineering. For example, bandwidth I-A and I-B may be more limited and those paths being non long-haul may not warrant the dual transmission.

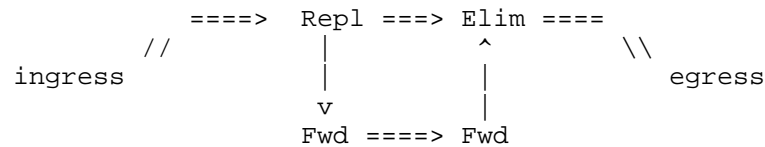


Figure 3: Enabled Adjacencies

- o For each of its bits that is set in the BIER header, the owner replication point resets the bit used for a copy and transmits towards the associated adjacency; to achieve this, the replication point copies the packet and inserts the relevant data plane information, such as next-hop label, MAC-address or source route header (for a BIER-TE routed adjacency), towards the adjacency that corresponds to the bit

Adjacency	BIER BitString
I->A	01011110
A->B	00011110
B->D	00010110
D->C	00010010
A->C	01001110

BitString in BIER Header as Packet Progresses

Table 2: BIER-TE in Action

- o Adversely, an elimination node on the path performs the Elimination Function which will remove duplicate packets (same flow-id, same sequence number) and performs a bitwise AND on the BitStrings from the various copies of the packet that it has received, before it forwards the packet with the resulting BitString. Details of the Elimination Function are described below.

Operation	BIER BitString
D->C	00010010
A->C	01001110
AND in C	00000010
C->E	00000000

BitString Processing at Elimination Point C

Table 3: BIER-TE in Action (cont.)

- o In this example, all the transmissions succeeded and the BitString at arrival has all the bits reset - note that the egress may be an Elimination Point in which case this is evaluated after this node has performed its AND operation on the received BitStrings).

Failing Adjacency	Egress BIER BitString
I->A	Frame Lost
I->B	Not Tried
A->C	00010000
A->B	01001100
B->D	01001100
D->C	01001100
C->E	Frame Lost
D->E	Not Tried

BitString indicating failures

Table 4: BIER-TE in Action (cont.)

- o But if a transmission failed along the way, one (or more) bit is never cleared. Table 4 provides the possible outcomes of a transmission. If the frame is lost, then it is probably due to a failure in either I->A or C->E, and the controller should enable I->B and D->E to find out. A BitString of 00010000 indicates unequivocally a transmission error on the A->C adjacency, and a BitString of 01001100 indicates a loss in either A->B, B->D or D->C; enabling D->E on the next packets may provide more information to sort things out.

In more details:

The BIER header is of variable size, and a DetNet network of a limited size can use a model with 64 bits if 64 adjacencies are enough, whereas a larger deployment may be able to signal up to 256 adjacencies for use in very complex paths. The format of this header is common to BIER and BIER-TE.

For the DetNet data plane, a replication point is an ingress point for more than one adjacency, and an elimination point is an egress point for more than one adjacency.

A pre-populated state in a replication node indicates which bits are served by this node and to which adjacency each of these bits corresponds. With DetNet, the state is typically installed by a controller entity such as a PCE. The way the adjacency is signaled in the packet is fully abstracted in the bit representation and must be provisioned to the replication nodes and maintained as a local state, together with the timing or shaping information for the associated flow.

The DetNet data plane uses BIER-TE to control which adjacencies are used for a given packet. This is signaled from the path ingress, which sets the appropriate bits in the BIER BitString to indicate which replication must happen.

The replication point clears the bit associated to the adjacency where the replica is placed, and the elimination points perform a logical AND of the BitStrings of the copies that it gets before forwarding.

As is apparent in the examples above, clearing the bits enables to trace a packet to the replication points that made any particular copy. BIER-TE also enables to detect the failing adjacencies or sequences of adjacencies along a path and to activate additional replications to counter balance the failures.

Finally, using the same BIER-TE bit for both directions of the steps of the ladder enables to avoid replication in both directions along the crossing adjacencies. At the time of sending along the step of the ladder, the bit may have been already reset by performing the AND operation with the copy from the other side, in which case the transmission is not needed and does not occur (since the control bit is now off).

5. Elimination Function (Normative)

This section defines the normative behavior of the Elimination Function with optional OAM sub-function.

The Elimination Function is performed logically on reception of BIER-TE packets. It is therefore not part of the adjacencies or otherwise assigned to a specific bit. "Logically" means that this specification does not constrain implementations, especially on multi-linecard/multi-chassis systems to perform EF on a physical egress module. It just implies that it has to happen before replication to the bits in the bitstring.

TBD: In addition to being an ingress, EF could as well be modelled as a new adjacency assigned to bits. The full adjacency of a bit could then be a sequence of EF followed by one (or more) of existing adjacencies. This is currently not considered by this document due to the lack of identified need to support this option - e.g.: problems that can not be equally/better be solved with EF logically on ingress.

The Elimination Function is more formally written as EF(OAM, BIFT, {flows}/*), and is configured like BIFTs from the BIER-TE controller host and/or other future mechanisms.

OAM is boolean and indicates whether OAM function of bitwise AND of received packet copies is performed. This OAM function requires additional memory/processing over EF without OAM. Note that the OAM function does not change the effect of the Elimination Function for BFR/receivers - they will continue to just receive the first copy of a packet. Instead, it will continue to track further copies solely for the purpose of providing OAM information. This also requires some timeout or sequence number advancement to decide when to terminate waiting for further copies of packets before considering the OAM analysis of this packet to be complete. BFR supporting this document SHOULD support the OAM sub-function.

BIFT indicates the <SD,SI,BSL> for which to perform EF. Devices SHOULD support enabling per EF. {flows}/* indicates the set of flows for which EF operates (using the specified BIFT). Duplicate elimination has to create per-flow state to remember which sequence number packets for this flow were already received. In the case of OAM also what bits were set in that received prior copy of the packet.

When a device supports "*", then it will automatically allocate such a flow-state for every new recognized flow and expire such flow state after an operator determined timeout of activity - for example with a default of 10 seconds. Dynamic allocation of flow-state may cause some initial duplicates before this state is working and it makes the BFR more vulnerable to state DOS attacks, but it will allow BIER applications to send flows with the benefit of EF without the help of the controller having to know and program every flow.

In the {flows} option, control procedures (e.g.: BIER-TE controller host) indicate to the BFR explicitly the set of flows for which it should install/operate the EF function. Note that the flow-id in the extended BIER encapsulation is the combination of BFIR-ID and entropy field of the BIER header.

BFR supporting this document MUST support the {flows} option and MAY support the "*" option.

The following picture explains the results of EF being performed on ingres in a typical example:

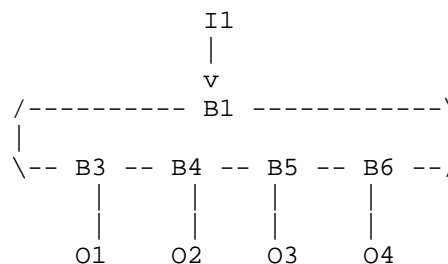


Figure 4: EF with Rings

Consider a simple ring where BFIR I1 generates BIER-TE packets. The bitstring indicates that the packet is sent hop-by-hop counterclockwise B1->B3->B4->B6 and counterclockwise B1->B6->B5->B4->B3. Bits for BFER O1, O2, O3 and O4 are also set. B3,B4,B5,B6,B7 perform EF. The result of this setup is that B2 creates two copies of the packets received from I1, one going to B6, the other to B3. Assume B4 first received the counter-clockwise copy from B3 and B5 the clockwise copy from B6. They will both forward these packets to each other because those were the first copies they saw, but they would block these second copies. Therefore only the link B4->B5 will have carried the packet copy twice (once in each direction). All the other ring links will only carry one copy of the packet.

This is notably different from schemes where EF is not performed before replication, but afterwards. In those schemes, both copies of the packets would flow counterclockwise around (most of) the ring, occupying more bandwidth.

6. Summary

With the addition of the functions of this document, BIER-TE becomes a potential option for the DetNet dataplane specifically beneficial when PREF (replication and elimination) is required for resilience

(to reduce packet loss). For DetNet multicast but also DetNet unicast. The unique capabilities of this approach are:

- o Explicit per-packet path selection for packet. Multicast and Unicast.
- o Control which replication take place on a per packet basis, so that replication points can be configured but not actually utilized
- o Trace the replication activity and determine which node replicated a particular packet
- o Measure the quality of transmission of the actual data packet along the replication segments and use that in a control loop to adapt the setting of the bits and maintain the reliability.

7. Implementation Status

A research-stage implementation of the forwarding plane for a 6TiSCH IOT use case was developed at Cisco's Paris Innovation Lab (PIRL) by Zacharie Brodard. It was implemented on OpenWSN Open-source firmware and tested on the OpenMote-CC2538 hardware. It implements the header types 15,16, 17, 18 and 19 (bit-by-bit encoding without group ID) in order to allow a BIER-TE protocol over IEEE802.15.4e.

This work was complemented with a Controller-based control loop by Hao Jiang. The controller builds the complex paths (called Tracks in 6TiSCH) and decides the setting of the BitStrings in real time in order to optimize the delivery ratio within a minimal energy budget.

Links:

github: <https://github.com/zach-b/openwsn-fw/tree/BIER>
OpenWSN firmware: <https://openwsn.atlassian.net/wiki/pages/viewpage.action?pageId=688187>
OpenMote hardware: <http://www.openmote.com/>

8. Security considerations

TBD.

9. IANA Considerations

This document has no IANA considerations.

10. Acknowledgements

The method presented in this document was discussed and worked out together with the DetNet Data Plane Design Team:

Jouni Korhonen
Janos Farkas
Norman Finn
Olivier Marce
Gregory Mirsky
Pascal Thubert
Zhuangyan Zhuang

The authors also like to thank the DetNet chairs Lou Berger and Pat Thaler, as well as Thomas Watteyne, 6TiSCH co-chair, for their contributions and support to this work.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

11.2. Informative References

- [I-D.dt-detnet-dp-alt]
Korhonen, J., Farkas, J., Mirsky, G., Thubert, P., Zhuangyan, Z., and L. Berger, "DetNet Data Plane Protocol and Solution Alternatives", draft-dt-detnet-dp-alt-04 (work in progress), September 2016.
- [I-D.ietf-bier-te-arch]
Eckert, T., Cauchie, G., Braun, W., and M. Menth, "Traffic Engineering for Bit Index Explicit Replication (BIER-TE)", draft-ietf-bier-te-arch-00 (work in progress), January 2018.
- [I-D.ietf-detnet-architecture]
Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-04 (work in progress), October 2017.
- [I-D.ietf-detnet-problem-statement]
Finn, N. and P. Thubert, "Deterministic Networking Problem Statement", draft-ietf-detnet-problem-statement-02 (work in progress), September 2017.

[I-D.ietf-detnet-use-cases]

Grossman, E., "Deterministic Networking Use Cases", draft-ietf-detnet-use-cases-14 (work in progress), February 2018.

[I-D.ietf-opsawg-oam-overview]

Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", draft-ietf-opsawg-oam-overview-16 (work in progress), March 2014.

[I-D.ietf-spring-segment-routing]

Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.

Authors' Addresses

Pascal Thubert (editor)
Cisco Systems
Village d'Entreprises Green Side
400, Avenue de Roumanille
Batiment T3
Biot - Sophia Antipolis 06410
FRANCE

Phone: +33 4 97 23 26 34
Email: pthubert@cisco.com

Toerless Eckert
Huawei USA - Futurewei Technologies Inc.
2330 Central Expy
Santa Clara 95050
USA

Email: tte+ietf@cs.fau.de

Zacharie Brodard
Ecole Polytechnique
Route de Saclay
Palaiseau 91128
FRANCE

Phone: +33 6 73 73 35 09
Email: zacharie.brodard@polytechnique.edu

Hao Jiang
Telecom Bretagne
2, rue de la Chataigneraie
Cesson-Sevigne 35510
FRANCE

Phone: +33 7 53 70 97 34
Email: hao.jiang@telecom-bretagne.eu

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: September 6, 2018

S. Venaas
M. Sivakumar
IJ. Wijnands
L. Ginsberg
Cisco Systems, Inc.
March 5, 2018

BIER MTU Discovery
draft-venaas-bier-mtud-00

Abstract

This document defines an IGP based mechanism for discovering the MTU of a BIER sub-domain. This document defines extensions to OSPF and IS-IS, but other protocols could potentially be extended. MTU discovery is usually done for a given path, while this document defines it for a sub-domain. This allows the computed MTU to be independent of the set of receivers. Also, the MTU is independent of rerouting events within the sub-domain.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

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

Table of Contents

1. Introduction	2
2. Conventions used in this document	3
3. IS-IS BIER MTU Sub-sub-TLV	3
4. OSPF BIER MTU Sub-TLV	4
5. IANA considerations	4
6. References	5
6.1. Normative References	5
6.2. Informative References	5
Authors' Addresses	5

1. Introduction

This document defines an IGP based mechanism for discovering the MTU of a BIER sub-domain. The discovered MTU indicates the largest possible BIER payload, such as an IP packet, that can be sent across any link in a BIER sub-domain. This is different from [I-D.ietf-bier-path-mtu-discovery] which performs Path MTU Discovery (PMTUD) for a set of receivers. PMTUD is based on probing, and when there are routing changes, e.g., a link going down, the actual MTU for a path may become less than was previously discovered, and there will be some delay until the next probe is performed. Also, the set of receivers for a flow may change at any time, which may cause the MTU to change. This document instead discovers a BIER sub-domain MTU, which is independent of paths and receivers within the sub-domain.

For convenience we will refer to an interface on a router as a BIER interface if the router has a BIER neighbor on the interface. That is, there is a directly connected router on that interface that is announcing a BIER prefix. We say that it is a BIER interface in a given sub-domain if the router itself announces a prefix tagged with the sub-domain, and there is BIER neighbor on the interface also announcing a prefix tagged with the sub-domain.

In order to allow MTU discovery in a BIER sub-domain, the procedure is as follows. Every BIER router, for each sub-domain with at least one local BIER interface in the sub-domain, per the above definition of a BIER interface, determines the largest payload that can be sent BIER encapsulated out of any of its BIER interfaces in the sub-domain. That is, for each local BIER interface in the sub-domain, it needs to determine the size of the largest BIER encapsulated payload

that can be sent out of that interface. We define the local sub-domain MTU of a router to be the minimum of the per BIER interface maximum payload size.

A BIER router announces a BIER prefix in either IS-IS or OSPF as specified in [I-D.ietf-bier-isis-extensions] and [I-D.ietf-bier-ospf-bier-extensions]. They both define a BIER Sub-TLV to be included with the prefix. There is one BIER Sub-TLV included for each sub-domain. This document defines how a router includes its local sub-domain MTU in each of the BIER Sub-TLVs it advertizes.

A router can discover the MTU of a BIER sub-domain by identifying all the prefixes that have a BIER Sub-TLV for the sub-domain. It then computes the minimum of the advertised MTU values for that sub-domain. This includes its local sub-domain MTU. This allows all the routers in the sub-domain to discover the same sub-domain wide MTU.

Note that a router should announce a new local MTU for a sub-domain immediately if the value becomes smaller than what it currently announces. This would happen if the MTU of an interface is configured to a smaller value, or the first BIER neighbor for a sub-domain is detected on an interface, and the MTU of the interface is less than all the other local BIER interfaces in the sub-domain. However, if BIER neighbors go away, or if an interface goes down, so that the local MTU becomes larger, a router SHOULD NOT immediately announce the larger value. A router MAY after some delay announce the new larger MTU. The intention is that dynamic events such as a quick link flap should not cause the announced MTU to be increased.

2. Conventions used in this document

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. IS-IS BIER MTU Sub-sub-TLV

A router uses the BIER MTU Sub-sub-TLV to announce the minimum BIER MTU of all its BIER enabled interfaces. The Sub-sub-TLV MUST be ignored if it is included multiple times.


```

      0                   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
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Type           |      Length      |           MTU           |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Type: TBD

Length: 2

MTU: MTU in octets

4. OSPF BIER MTU Sub-TLV

A router uses the BIER MTU Sub-TLV to announce the minimum BIER MTU of all its BIER enabled interfaces. It is a Sub-TLV of the BIER Sub-TLV, and SHOULD be included exactly once within each of the advertised BIER Sub-TLVs. The Sub-TLV MUST be ignored if it is included multiple times.

```

      0                   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
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Type           |      Length      |           MTU           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           MTU           |      Reserved      |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Type: TBD2

Length: 4

MTU: MTU in octets

5. IANA considerations

An allocation from the "sub-sub-TLVs for BIER Info sub-TLV" registry as defined in [I-D.ietf-bier-isis-extensions] is requested for the IS-IS BIER MTU Sub-sub-TLV. Please replace the string TBD in this document with the appropriate value.

An allocation from the "OSPF Extended Prefix sub-TLV" registry as defined in [RFC7684] is requested for the OSPF BIER MTU Sub-TLV. Please replace the string TBD2 in this document with the appropriate value.

6. References

6.1. Normative References

- [I-D.ietf-bier-isis-extensions]
Ginsberg, L., Przygienda, T., Aldrin, S., and Z. Zhang,
"BIER support via ISIS", draft-ietf-bier-isis-
extensions-09 (work in progress), February 2018.
- [I-D.ietf-bier-ospf-bier-extensions]
Psenak, P., Kumar, N., Wijnands, I., Dolganow, A.,
Przygienda, T., Zhang, Z., and S. Aldrin, "OSPF Extensions
for BIER", draft-ietf-bier-ospf-bier-extensions-15 (work
in progress), February 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W.,
Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute
Advertisement", RFC 7684, DOI 10.17487/RFC7684, November
2015, <<https://www.rfc-editor.org/info/rfc7684>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

6.2. Informative References

- [I-D.ietf-bier-path-mtu-discovery]
Mirsky, G., Przygienda, T., and A. Dolganow, "Path Maximum
Transmission Unit Discovery (PMTUD) for Bit Index Explicit
Replication (BIER) Layer", draft-ietf-bier-path-mtu-
discovery-03 (work in progress), January 2018.

Authors' Addresses

Stig Venaas
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: stig@cisco.com

Mahesh Sivakumar
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: masivaku@cisco.com

IJsbrand Wijnands
Cisco Systems, Inc.
De kleetlaan 6a
Diegem 1831
Belgium

Email: ice@cisco.com

Les Ginsberg
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: ginsberg@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2019

J. Xie
M. McBride
M. Chen
Huawei Technologies
L. Geng
China Mobile
July 2, 2018

Multicast VPN Using MPLS P2MP and BIER
draft-xie-bier-mvpn-mpls-p2mp-02

Abstract

MVPN is a widely deployed multicast service with mLDP or RSVP-TE P2MP as the P-tunnel. Bit Index Explicit Replication (BIER) is an architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. This document introduces a seamless transition mechanism from legacy MVPN using mLDP/RSVP-TE P2MP to MVPN using BIER by combining P2MP and BIER to form a P2MP based BIER as the P-tunnel. This will leverage the widely supported P2MP capability in both data-plane and control-plane, and will help introducing BIER in existing multicast networks to shift multicast delivery from MVPN using mLDP/RSVP-TE P2MP by two means: It is easier and more efficient for legacy routers to support BIER forwarding on the basis of widely supported P2MP forwarding, and it is more seamless for existing multicast networks to deploy BIER when some routers do not support BIER forwarding.

Requirements Language

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

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 3, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Applicability Statement	4
4. MVPN using P2MP based BIER	5
4.1. Overview	5
4.2. MVPN Transition from P2MP to P2MP based BIER	5
4.2.1. Use of the PTA in x-PMSI A-D Routes	6
4.3. Building P2MP based BIER forwarding state	8
5. P2MP based BIER Forwarding Procedures	8
5.1. Overview	8
5.2. P2MP based BIER forwarding	9
5.3. When Mid, Leaf or Bud nodes do not support P-CAPABILITY	11
5.4. When Leaf or Bud nodes do not support D-CAPABILITY	13
6. Provisioning Considerations	15
7. IANA Considerations	16
8. Security Considerations	16
9. Acknowledgements	16
10. References	17
10.1. Normative References	17
10.2. Informative References	18
Authors' Addresses	18

1. Introduction

[RFC6513] and [RFC6514] specify the protocols and procedures that a Service Provider (SP) can use to provide Multicast Virtual Private Network (MVPN) service to its customers. Multicast tunnels are created through an SP's backbone network; these are known as "P-tunnels". The P-tunnels are used for carrying multicast traffic across the backbone. The MVPN specifications allow the use of several different kinds of P-tunnel technology, such as mLDP P2MP and RSVP-TE P2MP. It is common for such a P-tunnel having a multicast-specific path.

Bit Index Explicit Replication (BIER) [RFC8279] is an architecture that provides optimal multicast forwarding through a "multicast domain", without requiring intermediate routers to maintain any per-flow state, by using a multicast-specific BIER header (per [RFC8296]).

[I-D.ietf-bier-mvpn] delivers a solution of MVPN using SPF based BIER defined in [RFC8279]. It can not, however, support a multicast-specific path well, something common in legacy MVPN deployment.

[RFC8279] provides a solution to support mid nodes without BIER-capability. It cannot, however, support deployment on a network that has edge nodes without BIER-capability, which may be common in some SP-networks, especially when most of the nodes in a network or part of a network are edge or service nodes.

This document introduces a seamless transition mechanism from legacy MVPN to MVPN using P2MP based BIER, by applying a BIER encapsulation in data-plane to eliminate per-flow states, while preserving existing features such as multicast-specific PATH.

It also introduces a seamless deployment solution on networks with Non-BIER-capability Edge nodes and/or Mid nodes, by exploring the P2MP/tree based BIER forwarding procedure in detail. Such a P2MP/tree based BIER is mentioned but not explored in detail in RFC8279.

2. Terminology

Readers of this document are assumed to be familiar with the terminology and concepts of the documents listed as Normative References. For convenience, some of the more frequently used terms and new terms list below.

- o LSP: Label Switch Path
- o LSR: Label Switching Router

- o P2MP: Point to Multi-point
- o P-tunnel: A multicast tunnel through the network of one or more SPs. P-tunnels are used to transport MVPN multicast data.
- o PMSI: Provider Multicast Service Interface
- o x-PMSI A-D route: a route that is either an I-PMSI A-D route or an S-PMSI A-D route.
- o PTA: PMSI Tunnel attribute. A type of BGP attribute known as the PMSI Tunnel attribute.
- o P2MP based BIER: BIER using P2MP LSP as topology
- o P-CAPABILITY: A capability to Process BitString in BIER Header of a packet.
- o D-CAPABILITY: A capability to Disposit BIER Header of a packet, including or excluding the BIER Label.
- o BSL: Bit String Length, that is 64, 128, 256, etc (per [RFC8279]).

3. Applicability Statement

The BIER architecture document [RFC8279] describes how each node forwards BIER packets hop by hop to neighboring nodes without generating duplicate packets. This forwarding is for the case where a form of underlay called "many to many " and built by IGP is used. Obviously, the case of underlay of "one to many" or P2MP is a simpler scenario, and the forwarding procedure naturally applies. However, as is well-known, such a forwarding procedure requires the support of hardware. The usage of the same forwarding method for both complex scenarios and simple scenarios will inevitably require complex hardware forwarding.

This document describes how BIER forwarding can be customized and simplified with an underlay of "one to many" or P2MP (see chapter 5). This customization and simplification eliminates some of the unnecessary data plane processing and so is easier to implement with existing hardware. Based on this customization of the forwarding method for P2MP-based BIER, a variety of Partial Deployment methods are given for the different capabilities of the hardware to support BIER forwarding. Compared with RFC8279, when there is no BIER forwarding capability on edge nodes, Partial Deployment can be carried out ; For the case where the intermediate node has no BIER forwarding capability, P2MP forwarding can be used without the need for unicast replication.

This document also describes a MVPN Transition solution that eliminates the per-flow state by introducing BIER MPLS encapsulation and forwarding in data-plane, while preserving the original control-plane protocol and its features, especially when some sort of path customizing being used. The said path customization include RSVP-TE P2MP using an explicit path, and MLDP P2MP where static route was used. These features can continue to retain, making the transition process seamless.

4. MVPN using P2MP based BIER

4.1. Overview

According to [RFC8279], the P2MP based BIER is a BIER which using a form of tree as the underlay. The P2MP LSP is not only a LSP, but also a topology as the BIER underlay. The P2MP based BIER is P-tunnel, which is used for bearing multicast flows. Every flow can be seen as binding to an independent tunnel, which is constructed by the BitString in the BIER header of every packet of the flow. Multicast flows are transported in SPMSI-only mode, on P2MP based BIER tunnels, and never directly on P2MP LSP tunnel.

Section 4.2 describes the overall principle of transitioning a Legacy MVPN using P2MP to a MVPN using BIER. It also describes the detail use of new types of PTA in BGP MVPN routes to indicate PEs to initialize the building of P2MP based BIER forwarding.

Section 4.3 describes the Underlay protocols to build P2MP based BIER forwarding briefly.

4.2. MVPN Transition from P2MP to P2MP based BIER

This section describes a MVPN transitioning solution that eliminates the per-flow state by introducing BIER MPLS encapsulation and forwarding procedure in data-plane, while preserving the originally deployed control-plane protocol and its features, especially when some sort of path customizing being used.

When transitioning a MVPN using mLDP P2MP P-tunnel, then continue using mLDP to build a P2MP based BIER forwarding, preserving the original mLDP features. For example, mLDP uses static route to specify a path other than the path of IGP.

When transitioning a MVPN using RSVP-TE P2MP P-tunnel, then continue using RSVP-TE to build a P2MP based BIER forwarding, preserving the original RSVP-TE features. For example, RSVP-TE use explicit path to specify a path other than the path of IGP.

4.2.1. Use of the PTA in x-PMSI A-D Routes

As defined in [RFC6514], the PMSI Tunnel attribute (PTA) carried by an x-PMSI A-D route identifies the P-tunnel that is used to instantiate a particular PMSI. If a PMSI is to be instantiated by P2MP LSP based BIER, the PTA is constructed by a BFIR, which is also an Ingress LSR. This document defines the following Tunnel Types:

+ TBD - RSVP-TE built P2MP BIER

+ TBD - mLDP built P2MP BIER

Allocation is expected from IANA for two new tunnel type codepoints from the "P-Multicast Service Interface Tunnel (PMSI Tunnel) Tunnel Types" registry. These codepoints will be used to indicate that the PMSI is instantiated by MLDP or RSVP-TE extension with support of BIER.

When the Tunnel Type is set to RSVP-TE built P2MP BIER, the Tunnel Identifier includes two parts, as follows:

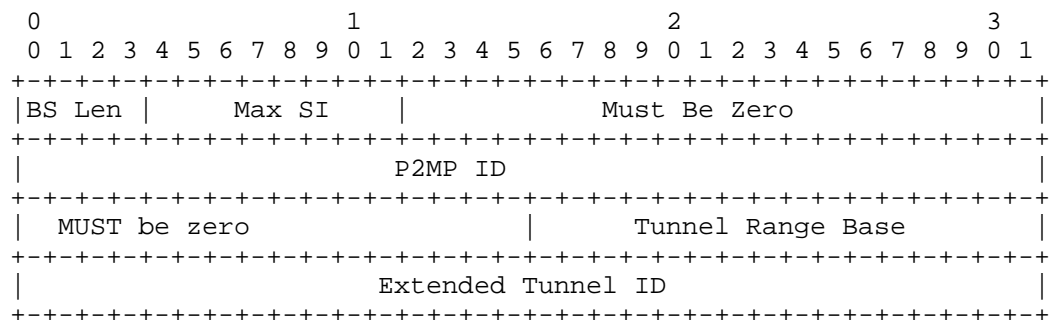


Figure 1: PTA of RSVP-TE built P2MP BIER

BS Len: A 4 bits field. The values allowed in this field are specified in section 2 of [RFC8296].

Max SI: A 1 octet field. Maximum Set Identifier (section 1 of [RFC8279]) used in the encapsulation for this BIER sub-domain.

<Extended Tunnel ID, Reserved, Tunnel Range Base, P2MP ID>: A ID as carried in the RSVP-TE P2MP LSP SESSION Object defined in [RFC4875].

The "Tunnel Range" is the set of P2MP LSPs beginning with the Tunnel Range base and ending with ((Tunnel Range base)+(Tunnel Number)- 1). A unique Tunnel Range is allocated for the BSL and a Sub-domain-ID implicated by the P2MP.

The size of the Tunnel Range is determined by the number of Set Identifiers (SI) (section 1 of [RFC8279]) that are used in the topology of the P2MP-LSP. Each SI maps to a single Tunnel in the Tunnel Range. The first Tunnel is for SI=0, the second Tunnel is for SI=1, etc.

When the Tunnel Type is set to mLDP built P2MP BIER, the Tunnel Identifier include two parts, as follows:

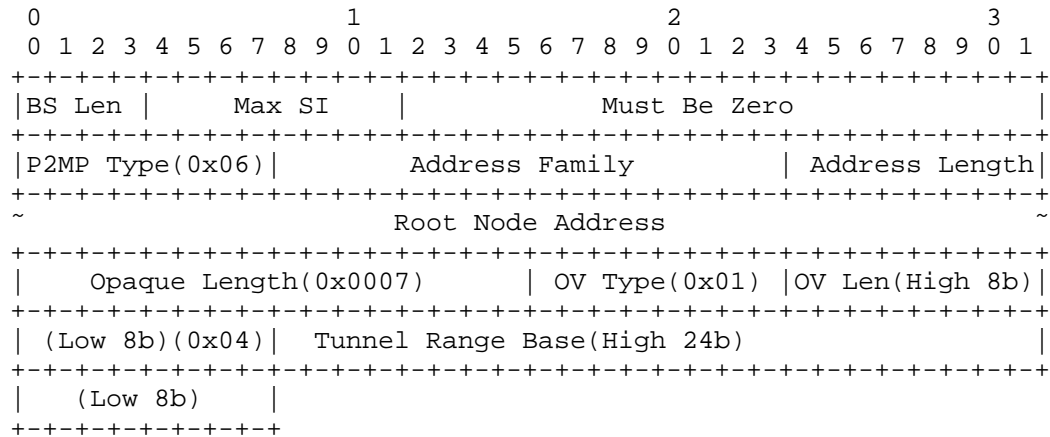


Figure 2: PTA of MLDP built P2MP BIER

BS Len: A 4 bits field. The values allowed in this field are specified in section 2 of [RFC8296].

Max SI: A 1 octet field. Maximum Set Identifier (section 1 of [RFC8279]) used in the encapsulation for this BIER sub-domain.

<Type=0x06, AF, AL, RootNodeAddr, Opqgue Length=0x0007, OV Type=0x01, OV Len=0x04, Tunnel Range Base>: A P2MP Forwarding Equivalence Class (FEC) Element, with a Generic LSP Identifier TLV as the opaque value element, defined in [RFC6388].

The "Tunnel Range" is the set of P2MP LSPs beginning with the Tunnel Range base and ending with ((Tunnel Range base)+(Tunnel Number)- 1). A unique Tunnel Range is allocated for the BSL and a Sub-domain-ID implicated by the P2MP.

The size of the Tunnel Range is determined by the number of Set Identifiers (SI) (section 1 of [RFC8279]) that are used in the topology of the P2MP-LSP. Each SI maps to a single Tunnel in the Tunnel Range. The first Tunnel is for SI=0, the second Tunnel is for SI=1, etc.

When the Tunnel Type is any of the above, The "MPLS label" field contain an upstream-assigned non-zero MPLS label. It is assigned by the router (a BFIR) that constructs the PTA. Absence of an MPLS Label is indicated by setting the MPLS Label field to zero.

When the Tunnel Type is any of the above, two of the flags, LIR and LIR-pF, in the PTA "Flags" field are meaningful. Details about the use of these flags can be found in [RFC6513], [I-D.ietf-bess-mvpn-expl-track] and [I-D.ietf-bier-mvpn]].

4.3. Building P2MP based BIER forwarding state

When P2MP based BIER are used, then it is not necessary to use IGP or BGP to build the BIER routing table and forwarding table. Instead, the BIER layer information is carried by MLDP or RSVP-TE, when they build the P2MP tree.

The detail procedure for building P2MP based BIER forwarding state using mLDP or RSVP-TE is outside the scope of this document.

5. P2MP based BIER Forwarding Procedures

5.1. Overview

This document specifies one OPTIONAL Forwarding Procedure of BIER encapsulation packet, on the condition that the BIER underlay topology is P2MP LSP, as describes in the above sections. It is in fact a customized forwarding procedure, and a detail exploration of BIER forwarding along a multicast-specific tree. Comparing to the common Forwarding Procedure described in [RFC8279], there is some considerable simplification:

1. Not need to Edit the BitString when forwarding packet to Neighbor, for the underlay P2MP topology is already loop-free and duplicate-free. This can further lead to a method to by-pass the BIER encapsulation packet when a node does not support the BitString process.
2. Not need to do a disposition function by parsing the BitString, for a P2MP can identify a disposition function by a node's Label when the P2MP is built. This can further reduce the complex BitString processing for legacy hardware on edge, and lead to a method to deploy on exist network when an edge node does not support BitString process.

The main principle of the optional forwarding procedure of the P2MP based BIER is, on the basis of P2MP forwarding procedure according to the BIER-MPLS label, to use the BitString to prune/filter the

undesired P2MP downstream. This is a smooth enhancement to the widely deployed P2MP forwarding, and easier to deploy on existing routers comparing to the many-to-many BIER forwarding.

The enhancement to the P2MP forwarding is to add a Forwarding BitMask to existing NHLFE defined in [RFC3031], for checking with the BitString in a packet, to determine whether the packet is to be forwarded or pruned. If the checking result by AND'ing a packet's BitString with the F-BM of the NHLFE (i.e., Packet->BitString &= F-BM) is non-zero, then forward the packet to the next-hop indicated by the NHLFE entry, and the Label is switched to the proper one in the NHLFE. If the result is zero, then do not forward the packet to the next-hop indicated by the NHLFE entry.

5.2. P2MP based BIER forwarding

For a P2MP tree, every node has a role of Root, Branch, Leaf, or Bud, as specified in [RFC4611].

EXAMPLE 1: Take the following figure as an example.

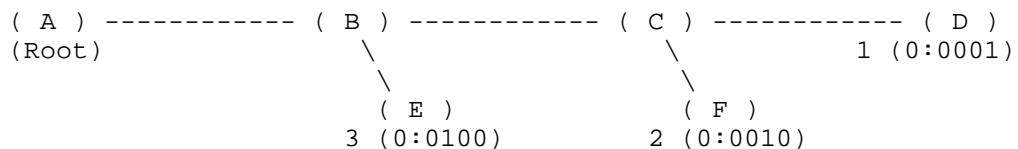


Figure 3: P2MP-based BIER Topology without BUD nodes

Forwarding Table on A:

- o NHLFE(TreeID, OutInterface<toB>, OutLabel<alloc by B>, F-BM<0111>)

Forwarding Table on C:

- o ILM(inLabel<alloc by C>, action<TreeID>, Flag=Branch|CheckBS, BSL)
- o NHLFE(TreeID, OutInterface<toD>, OutLabel<alloc by D>, F-BM<0001>)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>, F-BM<0010>)

For Node C, the ability to receive a MPLS-encapsulation BIER packet, match ILM and get a TreeID, replicate to NHLFE Entries of the TreeID according to the result of AND'ing the BitString of packet and the F-BM of a NHLFE Entry, is called a P-CAPABILITY, which means to Process BitString in each packet.

Forwarding Table on B is the same to C.

Forwarding Table on D:

- o ILM(inLabel<alloc by D>, action<TreeID>, Flag=Leaf|CheckBS, BSL)
- o LEAF(TreeID, F-BM<0001>, flag=PopBIERincluding)

When Node D receive a MPLS-encapsulation BIER packet, it get the Label and match ILM, then do a replication according to the LEAF and check whether to proceed by AND'ing the BitString in the replicated packet and the F-BM in the LEAF entry. When the AND'ing result is non-zero then do a POP to the packet to disposit the whole BIER header Including the BIER Label, which has a length of (12+BSL/8) octets.

Node D need to have a P-CAPABILITY, for it need to Process BitString in each packet to determin whether to replicate to a special LEAF, and then disposit the whole BIER header Including the BIER Label and forward the IP multicast packet further. Node D also need to do the disposition as well, which is called a D-CAPABILITY. D-CAPABILITY means to disposit the BIER header including or excluding the BIER Label in the begining. Here PopBIERincluding means pop the BIER header including the BIER Label, while PopBIERexcluding means pop the BIER header excluding the BIER Label.

Forwarding Tables on E and F are same to D.

Comparing to the forwarding procedure defined in [RFC8279], there are two benefits of using the customized P2MP based BIER forwarding:

1. Not need to walk every physical neighbor, but only need to walk downstream neighbors on a P2MP tree.
2. Not need to edit the BitString in every packet, but only need to swap the BIER Label.

EXAMPLE 2: Another example with P2MP BUD Nodes.

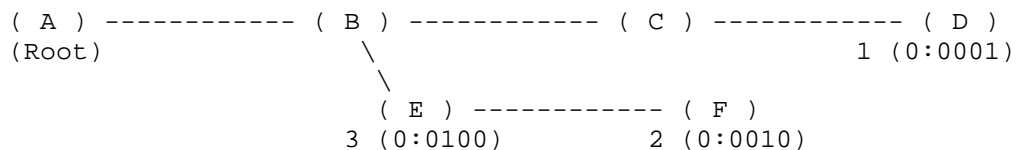


Figure 4: P2MP-based BIER Topology with BUD nodes

Forwarding Table on B (Branch Node):

- o ILM(inLabel<alloc by B>, action<TreeID>, Flag=Branch|CheckBS, BSL)

- o NHLFE(TreeID, OutInterface<toE>, OutLabel<alloc by E>, F-BM<0110>)
- o NHLFE(TreeID, OutInterface<toC>, OutLabel<alloc by C>, F-BM<0001>)

Node B, which is a Branch Node, only need to use its P-CAPABILITY.

Forwarding Table on E (BUD Node):

- o ILM(inLabel<alloc by E>, action<TreeID>, Flag=Bud|CheckBS, BSL)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>, F-BM<0010>)
- o LEAF(TreeID, F-BM<0100>, flag=PopBIERincluding)

When Node E receive a MPLS-encapsulation BIER packet, it get the Label and match ILM, then do a replication according to the NHLFEs and check whether to proceed by AND'ing the BitString in the replicated packet and the F-BM in the NHLFE/LEAF entry. When the AND'ing result is non-zero for the second LEAF then do a POP to the packet to disposit the whole BIER header, which has a length of (12+BSL/8) octets.

Node E, which is a BUD Node, has both the two capacities: P-CAPABILITY and D-CAPABILITY. P-CAPABILITY is need to be used for every NHLFE/LEAF, and D-CAPABILITY is need for the NHLFE that has a PopBIERincluding flag.

5.3. When Mid, Leaf or Bud nodes do not support P-CAPABILITY

The procedures of Section 5.2 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. In this section, we describe procedures that can be used if the routing underlay is a P2MP tree with BIER information in the domain.

For a P2MP tree, every node has a role of Root, Branch, Leaf, or Bud. The role is determined when the tree is built. The method is suitable for conditions when Mid, Leaf or Bud nodes do not support P-CAPABILITY.

EXAMPLE 1: Take Figure 4 as an example.

If D, F, E support BIER, and C don't support BIER, then we can configure on C to indicate it to use P2MP for BIER packets forwarding. Then C build a P2MP forwarding entry, while still pass the BIER information in control-plane. For example, D send a P2MP FEC Mapping message to C with a BitMask 0001, F send a P2MP FEC

Mapping message to C with a BitMask 0010, and C send a P2MP FEC Mapping message to B with a BitMask, but C build a P2MP forward entry like this:

- o ILM(inLabel<alloc by C>, action<TreeID>, Flag=Branch)
- o NHLFE(TreeID, OutInterface<toD>, OutLabel<alloc by D>)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>)

If D don't support BIER P-CAPABILITY, but it support BIER D-CAPABILITY, then the above method is still valid.

Forwarding Table on D when D don't have a P-CAPABILITY:

- o ILM(inLabel<alloc by D>, action<TreeID>, Flag=Leaf, BSL)
- o NHLFE(TreeID, flag=PopBIERincluding)

When Node D receive a MPLS-encapsulation BIER packet, it get the Label and match ILM, then do a replication according to the NHLFE but don't do the check by AND'ing the BitString in the replicated packet and the F-BM in the NHLFE entry. And then do a POP to the packet to disposit the whole BIER header, which has a length of (12+BSL/8) octets.

Another alternative form of Forwarding Table on D can also be the following when D don't have a P-CAPABILITY:

- o ILM(inLabel<alloc by D>, action<PopBIERincluding>, Flag=Leaf, BSL)

When Node D receive a MPLS-encapsulation BIER packet, it get the Label and match ILM, then do a POP action according to the ILM to pop the whole (12+BSL/8) octets from the Label position.

EXAMPLE 2: Take BUD Node E in Figure 5 as another example.

Forwarding Table on Bud Node E when E don't have a P-CAPABILITY:

Forwarding Table on E when E don't have a P-CAPABILITY:

- o ILM(inLabel<alloc by E>, action<TreeID>, Flag=Bud, BSL)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>)
- o LEAF(TreeID, flag=PopBIERincluding)

One can see that, this method can support widely Non-BIER Nodes in a network, no matter the node has a Mid, Leaf or Bud role, and would never result in any ingress-replication through unicast tunnel, which may cause a overload on a link.

One can also see that, [RFC8279] only support Non BIER-capability nodes being the Mid nodes, and never allow a BFER nodes to be Non BIER-capability.

5.4. When Leaf or Bud nodes do not support D-CAPABILITY

A more tolerant variant of the above, when Leaf or Bud nodes do not support D-CAPABILITY, would be the following:

EXAMPLE 1: Take Figure 4 as an example.

If D even don't support BIER P-CAPABILITY or D-CAPABILITY, then POP the whole BIER Header except the first four octets Label field of a packet before it come to D. This requires C to build a forwarding table like this:

Forwarding Table on C (Branch Node):

- o ILM(inLabel<alloc by E>, action<TreeID>, Flag=Branch|CheckBS, BSL)
- o NHLFE(TreeID, OutInterface<toD>, OutLabel<alloc by D>, F-BM<0001>, Flag=PopBIERexcluding)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>, F-BM<0010>)

The Flag PopBIERexcluding means POP the BIER Header excluding the first 4 octets BIER Label in a packet, that is a Length of (8+BSL/8)

If D don't support BIER P-CAPABILITY or D-CAPABILITY, and C don't support BIER P-CAPABILITY, then it requires B to build a forwarding table, to ensure the BIER Header except the first four octets Label field of a packet is popped before replicated to C, and requires C to build a forwarding table of a pure P2MP branch, and requires F to build a forwarding table of a pure P2MP Leaf. Their forwarding tables are like below:

Forwarding Table on B (Branch Node):

- o ILM(inLabel<alloc by B>, action<TreeID>, Flag=Branch|CheckBS, BSL)
- o NHLFE(TreeID, OutInterface<toC>, OutLabel<alloc by C>, F-MB<0011>, Flag=PopBIERexcluding)

- o NHLFE(TreeID, OutInterface<toE>, OutLabel<alloc by E>, F-BM<0100>)

Forwarding Table on C (Branch Node):

- o ILM(inLabel<alloc by C>, action<TreeID>, Flag=Branch)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>)

Forwarding Table on D (Branch Node):

- o ILM(inLabel<alloc by D>, action<PopLabel>, Flag=Leaf)

Here PopLabel mean to pop the Label, which is in fact a P2MP LSP Label. It is a basic capability of any LSR.

Forwarding Table on F (Branch Node):

- o ILM(inLabel<alloc by F>, action<PopLabel>, Flag=Leaf)

Here PopLabel mean to pop the Label, which is in fact a P2MP LSP Label. It is a basic capability of any LSR, and the Forwarding table on F is in fact a P2MP one.

Note that, although F support BIER, which means it can deal with a BIER packet, but it must downshift its forwarding table to a pure P2MP one, because the packet it received doesn't include a BIER Header but a P2MP Label packet due to the POP behaving of its upstream node.

EXAMPLE 2: Take Figure 5 as another example.

If E even don't support BIER P-CAPABILITY or D-CAPABILITY, then POP the whole BIER Header Except the first four octets Label field of a packet before it come to D. This requires B to build a forwarding table like this:

Forwarding Table on B (Branch Node):

- o ILM(inLabel<alloc by B>, action<TreeID>, Flag=Branch|CheckBS, BSL)
- o NHLFE(TreeID, OutInterface<toC>, OutLabel<alloc by C>, F-MB<0011>)
- o NHLFE(TreeID, OutInterface<toE>, OutLabel<alloc by E>, F-BM<0100>, Flag=PopBIERexcluding)

Forwarding Table on E (Bud Node):

- o ILM(inLabel<alloc by E>, action<TreeID>, Flag=Bud)
- o NHLFE(TreeID, OutInterface<toF>, OutLabel<alloc by F>)
- o LEAF(TreeID, flag=PopLabel)

Forwarding Table on F (Branch Node):

- o ILM(inLabel<alloc by F>, action<PopLabel>, Flag=Leaf)

Note that, although F support BIER, which means it can deal with a BIER packet, but it must downshift its forwarding table to a pure P2MP Leaf, because the packet it received doesn't include a BIER Header but a P2MP Label packet due to the POP behaving of its upstream node.

One can see that, when some Leaf or Bud nodes even don't have a D-CAPABILITY, we can do a POP action to disposing the BIER header excluding the BIER Label in the begining before the packet arrive the node. This is similar to a Penultimate Hop Popping in a P2P LSP.

6. Provisioning Considerations

P2MP based BIER use concepts of both P2MP and BIER. Some provisioning considerations list below:

Sub-domain:

In P2MP based BIER, every P2MP is a specific BIER underlay topology, and an implicit Sub-domain. RSVP-TE/MLDP build the BIER information of the implicit sub-domain when building the P2MP tree. MVPN get the implicit sub-domain by provisioning.

BFR-prefix:

In P2MP LSP based BIER, every BFR is also a LSR. So the BFR-prefix in the sub-domain is by default identified by LSR-id. Additionally, When BFR/LSR is also a MVPN PE, BFR-prefix is also the same as Originating Router's IP Address of x-PMSI A-D route or Leaf A-D route.

BFR-id:

When using protocols like RSVP-TE, which initializes P2MP LSP from a specific Ingress Node, BFR-id which is unique in P2MP LSP scope, can be auto-provisioned by Ingress Node, or conventionally configure on every Egress Nodes.

BSL and BIER-MPLS Label Block Size:

In P2MP LSP based BIER, Every P2MP LSP or implicit sub-domain requires a single BSL, and a specific BIER-MPLS Label block size for this BSL.

VPN-Label:

The P2MP based BIER 'P-tunnel' can be shared by multiple VPNs or a single VPN. When a P2MP based BIER being shared by multiple VPNs, an Upstream-assigned VPN-Label is required. It can be auto-provisioned or manual configured by the BFIR or Ingress LSR.

In fact, [RFC6513] has defined the method of "Aggregating Multiple MVPNs on a Single P-Tunnel". But unfortunately it is not widely deployed because of the serious trade-off between state saving and bandwidth waste. The BIER encapsulation and forwarding method give it a chance to eliminate the trade-off while gaining a completely state saving.

Even when such an aggregating is not used, it is still adequate to use BIER to save state by sharing one P2MP based BIER "P-tunnel" for multi flows in one specific VPN.

For seamless transitioning from legacy MVPN deployment and existing network, it is recommended not to use such an aggregating, as well as to use such an aggregating.

7. IANA Considerations

Allocation is expected from IANA for two new tunnel type codepoints for "RSVP-TE built P2MP based BIER" and "MLDP built P2MP based BIER" from the "P-Multicast Service Interface Tunnel (PMSI Tunnel) Tunnel Types" registry.

8. Security Considerations

This document does not introduce any new security considerations other than already discussed in [RFC8279].

9. Acknowledgements

The authors would like to thank Eric Rosen, Tony Przygienda, IJsbrand Wijnands and Toerless Eckert for their thoughtful comments and kind suggestions.

10. References

10.1. Normative References

- [I-D.ietf-bess-mvpn-expl-track]
Dolganow, A., Kotalwar, J., Rosen, E., and Z. Zhang,
"Explicit Tracking with Wild Card Routes in Multicast
VPN", draft-ietf-bess-mvpn-expl-track-09 (work in
progress), April 2018.
- [I-D.ietf-bier-mvpn]
Rosen, E., Sivakumar, M., Aldrin, S., Dolganow, A., and T.
Przygienda, "Multicast VPN Using BIER", draft-ietf-bier-
mvpn-11 (work in progress), March 2018.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol
Label Switching Architecture", RFC 3031,
DOI 10.17487/RFC3031, January 2001,
<<https://www.rfc-editor.org/info/rfc3031>>.
- [RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S.
Yasukawa, Ed., "Extensions to Resource Reservation
Protocol - Traffic Engineering (RSVP-TE) for Point-to-
Multipoint TE Label Switched Paths (LSPs)", RFC 4875,
DOI 10.17487/RFC4875, May 2007,
<<https://www.rfc-editor.org/info/rfc4875>>.
- [RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B.
Thomas, "Label Distribution Protocol Extensions for Point-
to-Multipoint and Multipoint-to-Multipoint Label Switched
Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011,
<<https://www.rfc-editor.org/info/rfc6388>>.
- [RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/
BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February
2012, <<https://www.rfc-editor.org/info/rfc6513>>.
- [RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP
Encodings and Procedures for Multicast in MPLS/BGP IP
VPNs", RFC 6514, DOI 10.17487/RFC6514, February 2012,
<<https://www.rfc-editor.org/info/rfc6514>>.
- [RFC6625] Rosen, E., Ed., Rekhter, Y., Ed., Hendrickx, W., and R.
Qiu, "Wildcard in Multicast VPN Auto-Discovery Routes",
RFC 6625, DOI 10.17487/RFC6625, May 2012,
<<https://www.rfc-editor.org/info/rfc6625>>.

- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.
- [RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <<https://www.rfc-editor.org/info/rfc8296>>.

10.2. Informative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

Authors' Addresses

Jingrong Xie
Huawei Technologies

Email: xiejingrong@huawei.com

Mike McBride
Huawei Technologies

Email: mmcbride7@gmail.com

Mach Chen
Huawei Technologies

Email: mach.chen@huawei.com

Liang Geng
China Mobile
Beijing 100053

Email: gengliang@chinamobile.com

BIER WG
Internet-Draft
Intended status: Informational
Expires: September 7, 2019

Quan Xiong
Greg Mirsky
ZTE Corporation
Fangwei Hu
Individual
March 6, 2019

The Resilience for BIER
draft-xiong-bier-resilience-02.txt

Abstract

Bit Index Explicit Replication (BIER) is an architecture for the forwarding of multicast data packets. In some scenarios, the resilience should be provided to guarantee the multicast data is protected by a given backup resource and forwarded successfully to the receivers in BIER-specific network.

This document discusses the resilience use cases, requirements and proposes solutions for BIER, including the protection and restoration mechanisms and detection methods.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 7, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
1.2. Terminology	3
2. BIER Resilience Use Cases	3
2.1. BIER End-to-End 1+1 Protection	3
2.2. BIER End-to-End Restoration	4
2.3. BIER Link Protection	5
3. Management and Control Considerations	6
4. Security Considerations	6
5. IANA Considerations	6
6. Acknowledgements	6
7. References	6
7.1. Normative References	6
7.2. Informational References	7
Authors' Addresses	7

1. Introduction

[RFC8279] defined Bit Index Explicit Replication (BIER) architecture as a solution for the forwarding of multicast data packets. The routers which support BIER are known as Bit-Forwarding Router (BFR) and the multicast data packet enters a BIER domain at a Bit-Forwarding Ingress Router (BFIR) and leaves at one or more Bit-Forwarding Egress Routers (BFERs).

[I-D.eckert-bier-te-frr] provides some protection mechanisms for traffic engineering in a BIER domain. However, there is no mechanism to protect multicast traffic against BIER-specific network failures. In some scenarios, the resilience should be provided to guarantee the multicast data is protected by a given backup resource and forwarded successfully to the receivers in BIER-specific network.

This document describes the resilience use cases and requirements for BIER-specific network and discusses the protection and restoration mechanisms and detection methods.

1.1. Requirements Language

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

1.2. Terminology

The terminology is defined as [RFC8279].

2. BIER Resilience Use Cases

The resilience use cases for a BIER-specific network should be considered including end-to-end and link protection scenarios. The protection, restoration, and related detection mechanisms MUST be provided for BIER resilience against a failure of a link or a node.

2.1. BIER End-to-End 1+1 Protection

The end-to-end protection mechanisms for a BIER-specific network should be considered in some scenarios like shown in Figure 1. It includes end-to-end 1+1 protection and restoration use cases. Two disjoint end-to-end multicast paths that are available for 1+1 protection or restoration from BFIR to BFERs should be provided. One path could be BFIR->BFR1->BFR2->BFR3->BFER1 and BFIR->BFR1->BFR2->BFR3->BFER2; and the alternative path is BFIR->BFR6->BFR5->BFR4->BFER1 and BFIR->BFR6->BFR5->BFR4->BFER2.

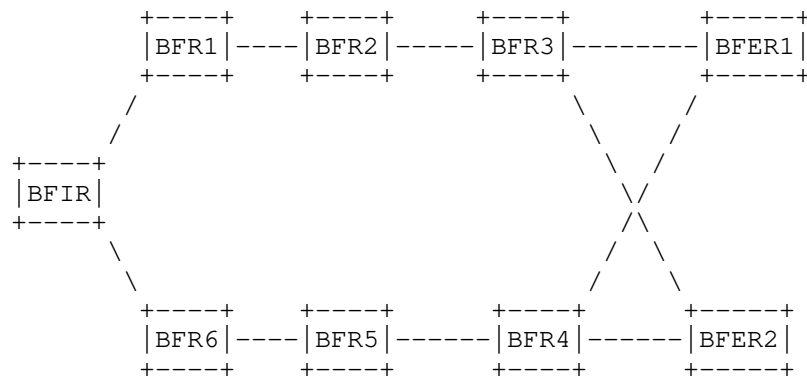


Figure 1: BIER End-to-End Protection and Restoration

For a 1+1 protection scenario, it is referred to as live-live, the BFIR sends two flows of multicast traffic to all BFERs through the

disjunct multipoint paths. BFERs need to merge the two flows when no failure happens. The BFERs MUST monitor and detect multicast failures and switch from one flow to another when a failure of a flow is detected.

For example, in a Deterministic Networking (DetNet) service, Packet Replication Function (PRF) is used in combination with Packet Elimination Function (PEF) and usually referred to as PREF. PREF is used in DetNet to lower the packet loss metric and it can be viewed as an example of live-live terminated within BIER domain. PRF replicates packets into multiple DetNet member flows and sends them along multiple different paths to the destinations and PEF eliminates the duplication based on the failure detection.

The failure detection mechanism for the end-to-end 1+1 protection scenario MUST be able to monitor and detect multicast failures in each working path. P2MP BFD [I-D.ietf-bfd-multipoint] MAY be used to verify multipoint connectivity between a BFIR and a set of BFERs. [I-D.hu-bier-bfd] describes the use of p2mp BFD in a BIER domain.

End-to-end 1+1 protection provides fast switch but low resource utilization. All BFERs MAY receive two copies from two paths in the no-failure scenario and the receivers MUST be able to choose one of them and eliminate the duplication.

2.2. BIER End-to-End Restoration

This section discusses the end-to-end restoration for BIER. If duplicate transmission is not desirable for some networks, the restoration mechanism may be taken into consideration where only one copy is sent to each receiver. The BFIR will send multicast flows onto the original path. If the BFIR detects a failure in the multicast path, the BFIR MAY create a new multicast tree and switch the multicast flow accordingly.

The failure detection mechanism for end-to-end restoration use case MUST be enable receivers (tails) to monitor and detect multicast failures in the multicast tree and notify the head node. BIER-specific extensions MAY be proposed based on [I-D.ietf-bfd-multipoint-active-tail]. The P2MP active tail detection method extends the mechanism defined in [I-D.ietf-bfd-multipoint]. It allows tails to notify the head of the failure of the multicast path and can be used in multipoint and multicast networks, e.g., in BIER domain as described in [I-D.hu-bier-bfd].

If P2MP BFD uses the active tail mode, then when one of the BFERs detects the failure, it will send a message to the BFIR. The BFIR

will create a new multicast path to restore the service and notify BFERs of switchover and start forwarding the multicast flows over the restoration path.

2.3. BIER Link Protection

Local protection, i.e., link or node protection, MAY be considered for BIER domain as an alternative to end-to-end protection. The nodes which are the BFRs in BIER network and they exchange the information needed for them to forward packets to each other using BIER. The node protection MAY be provided by using mechanisms already existing in the underlay network, for example, described in [I-D.eckert-bier-te-frr].

A BFR MAY send BIER packets to directly connected BIER neighbors through a BIER link without requiring a routing underlay. Link protection SHOULD be considered in BIER domain. The detection of link failure MAY use the Point-to-Point BFD detection defined in [RFC5880]. A set of extension for BIER-specific P2P BFD SHOULD be proposed in further discussion.

As shown in Figure 2, the BIER path from BFIR to BFERs is BFIR->BFR4->BFR3->BFR2->BFER1 and BFIR->BFR4->BFR3->BFER2. If the BIER link from BFR4 to BFR3 fails, the failure can be protected by the backup paths over BFR4->BFR1->BFR2->BFR3.

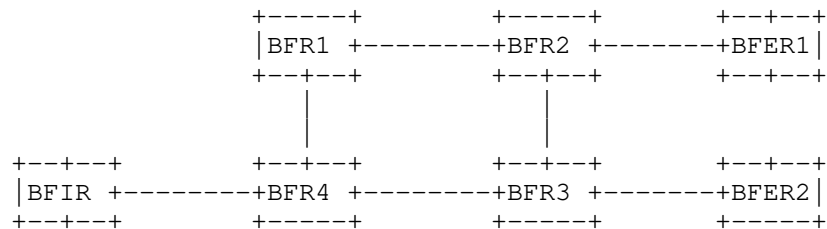


Figure 2: BIER Link Protection

As discussed in [I-D.eckert-bier-te-frr], the BIER link protection MAY use the existing RSVP-TE/P2MP or SR tunnel bypass. When a node detects a failure on a link, it MAY be assumed that the link has failed and the traffic is switched onto the pre-established backup path to get packets to the downstream node.

Also, as discussed in [RFC7490], the Topology Independent Loop-free Alternate Fast Re-route (TI-LFA) Fast Reroute (FRR) approach that achieves guaranteed coverage against link or node failure in the

Interior Gateway Protocol (IGP) network MAY be applied in BIER network.

3. Management and Control Considerations

BIER protection or restoration configuration, including BIER end-to-end protection, restoration, link/node protection and related information, MAY be defined and controlled from a centralized controller or a network management system. A failure detection and notification mechanism MUST be supported. The fast protection switching MUST be supported to minimize the loss of BIER packets due to BIER network failure.

4. Security Considerations

Security aspects of protection in BIER domain may be considered in relation to the data plane, and handling the dedicated OAM packets used to detect, signal a failure, coordinate the state in the BIER protection domain.

5. IANA Considerations

TBD

6. Acknowledgements

Authors would like to thank the comments and suggestions from Jeffrey (Zhaohui) Zhang.

7. References

7.1. Normative References

[I-D.hu-bier-bfd]

Xiong, Q., Mirsky, G., hu, f., and C. Liu, "BIER BFD", draft-hu-bier-bfd-03 (work in progress), February 2019.

[I-D.ietf-bfd-multipoint]

Katz, D., Ward, D., Networks, J., and G. Mirsky, "BFD for Multipoint Networks", draft-ietf-bfd-multipoint-19 (work in progress), December 2018.

[I-D.ietf-bfd-multipoint-active-tail]

Katz, D., Ward, D., Networks, J., and G. Mirsky, "BFD Multipoint Active Tails.", draft-ietf-bfd-multipoint-active-tail-10 (work in progress), November 2018.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <<https://www.rfc-editor.org/info/rfc7490>>.
- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.

7.2. Informational References

- [I-D.eckert-bier-te-frr]
Eckert, T., Cauchie, G., Braun, W., and M. Menth,
"Protection Methods for BIER-TE", draft-eckert-bier-te-frr-03 (work in progress), March 2018.

Authors' Addresses

Quan Xiong
ZTE Corporation
No.6 Huashi Park Rd
Wuhan, Hubei 430223
China

Phone: +86 27 83531060
Email: xiong.quan@zte.com.cn

Greg Mirsky
ZTE Corporation
USA

Email: gregimirsky@gmail.com

Fangwei Hu
Individual
China

Email: hufwei@gmail.com

BIER
Internet-Draft
Intended status: Standards Track
Expires: January 13, 2021

Z. Zhang
ZTE Corporation
B. Wu
Individual
Z. Zhang
Juniper Networks
IJ. Wijnands
Cisco Systems, Inc.
Y. Liu
China Mobile
July 12, 2020

BIER Prefix Redistribute
draft-zwzw-bier-prefix-redistribute-07

Abstract

This document defines a BIER proxy function to interconnect different underlay routing protocol areas in a network. And a new BIER proxy range sub-TLV is also defined to convey BIER BFR-id information across the routing areas.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 13, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Problem statement	2
2. Proposal	5
3. Advertisement	6
3.1. BIER proxy range sub-TLV	7
3.2. Proxy range sub-TLV usage	9
4. IANA Considerations	10
5. Security Considerations	10
6. Acknowledgements	10
7. References	10
7.1. Normative References	10
7.2. Informative References	11
Authors' Addresses	12

1. Problem statement

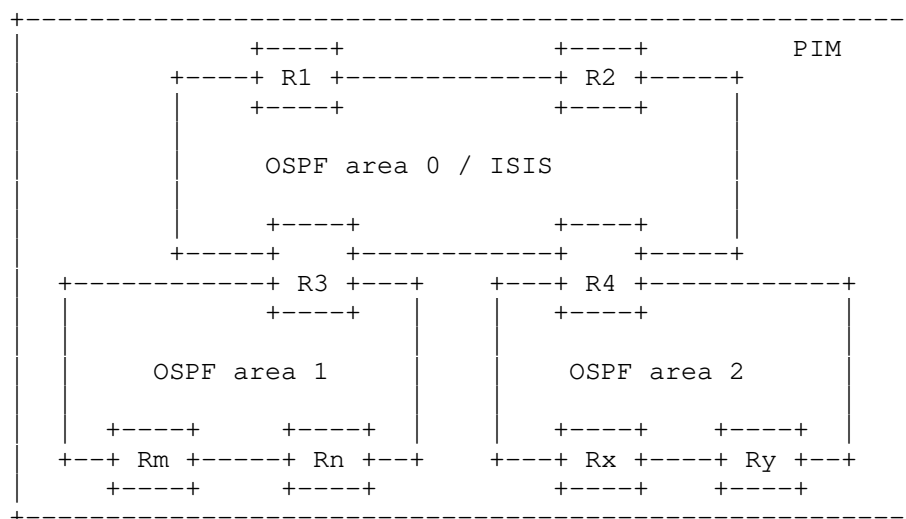


Figure 1

Figure 1 shows that there are three areas in a traditional network. In some deployment situations, different routing protocols may also be used in the network. There are just small amount of routers in each area. Currently, multicast services are provided in this kind of network by using protocol independent feature of PIM [RFC7761].

BIER could be a candidate multicast protocol to replace PIM to reduce multicast states in the network. BIER [RFC8279] is a new architecture for the forwarding of multicast data packets. It does not require a protocol for explicitly building multicast distribution trees, nor does it require intermediate nodes to maintain any per-flow state. In order to build BIER forwarding plane, BIER key parameters must be flooded in one BIER domain such as BFR-prefix, BFR-id, subdomain-id, and so on. The routing protocols which are used to flood these BIER parameters are called BIER routing underlay. The associated routing protocol extensions are defined in documents such as [RFC8401], [RFC8444], [I-D.ietf-bier-idr-extensions], [I-D.ietf-bier-ospfv3-extensions], and so on.

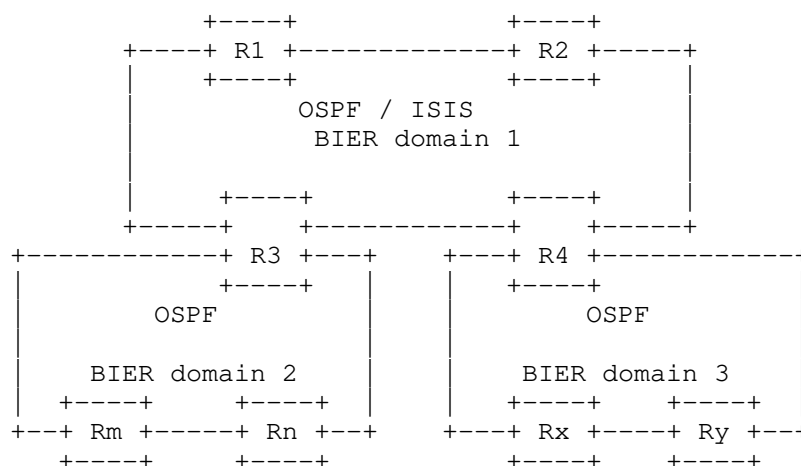


Figure 2

Based on the BIER design, a BIER domain is limited by the underlay routing protocols flooding scope. In case we want deploy BIER instead of PIM, there are several BIER domains because of different underlay routing areas limitation. Multiple encapsulating/decapsulation executions are needed to across multiple BIER domains. These executions slow down the forwarding efficiency. The border routers also need to maintain overlay state, which is undesired.

For example in figure 2, suppose that R1 and R2 connect with multicast source. Rm, Rn, Rx and Ry connect with multicast receivers. According the existed advertisement method defined in [RFC8401], [RFC8444], and so on, in BIER domain 1, R1, R2, R3 and R4 are BIER edge routers. In BIER domain 2, R3 and Rm, Rn are BIER edge routers. In BIER domain 3, R4 and Rx, Ry are BIER edge routers.

R1/R2 encapsulates BIER packet when multicast flows come into this BIER domain, R3/R4 decapsulates the BIER packet, and encapsulates them again, sends them into BIER domain 2/3. Rm, Rn, Rx and Ry decapsulates the packets and forwards them to receivers.

Due to the decapsulation and encapsulation execution in R3 and R4, the forwarding efficiency is slowed down, especially when there are not large amount of routers in each BIER domain.

Section 2.3 in [RFC8444] defines the duplication function across OSPF areas. Except the homogenization network, there is the hybrid network showed in figure 2 that several areas formed by different IGP protocols, and they are need to be merged into one BIER domain. In the hybrid network, necessary advertisement transform need to be

used. And further, necessary optimization method can be used to reduce the amount of the advertisement items.

2. Proposal

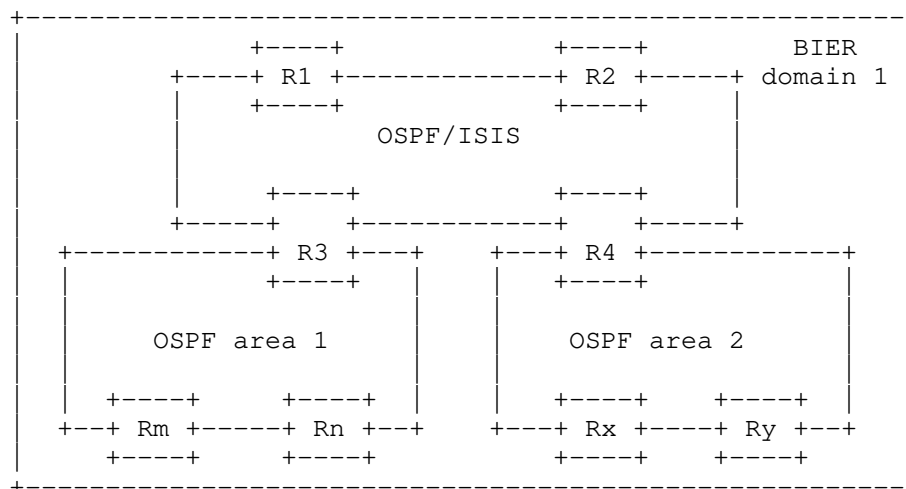


Figure 3

It is more efficient to deploy BIER by creating one BIER domain for the hybrid network to achieve forwarding benefit.

Since the limitation of the BIER routing protocol scope, BFR-id is confined to only one routing area. A BIER proxy function is introduced to transport BIER BFR-id information in a BIER domain across multiple routing protocol areas. So BIER forwarding tables can be built across multiple underlay routing protocols to replace encapsulation/ decapsulation processing. In the current deployment, border router (ABR) has a similar role, ABR summaries unicast routing information from one routing protocol area and sends it to another routing area by new routing protocol messages. So ABR can implement BIER proxy function to summarize BIER BFR-id information from one routing protocol area and sends it to another routing area.

In figure 3, R3/R4 connects two areas which running same or different routing protocols, they can be used as BIER proxies to transport BIER information. For example, after R3 receives BFR-ids information from OSPF area 1 and sends it to ISIS routing area (the upper area), the routers in ISIS routing area can generate BIER forwarding items toward the BFR-ids in OSPF area 1 through R3. Similarly, R3 receives BFR-ids information from the upper area and sends them into OSPF area 1, the routers in OSPF area 1 can build BIER forwarding items toward

the BFR-ids in ISIS area. R4 does the same function, the BIER forwarding plane is constructed accordingly.

For example, in this network, suppose that Rm and Rn have the prefix of 201.1.1.1/32, 201.1.1.2/32. In order to build one BIER domain which includes these three IGP areas, R3 advertises the BFR-ids of Rm/Rn with associated prefixes (201.1.1.1/32, 201.1.1.2/32) into the upper area. Similarly, R4 advertises the BFR-ids of Rx/Ry with associated prefixes (202.1.1.1/32, 202.1.1.2/32) into the upper area too.

And R3/R4 advertises the prefixes of R1 and R2 (suppose that the prefixes are 200.1.1.1/32 and 200.1.1.2/32) with associated BFR-ids into IGP area 1 and area 2. Also, R3 advertises the prefixes learned from R4 (202.1.1.1/32, 202.1.1.2/32) with associated BFR-ids into IGP area 1. R4 also advertises the prefixes (201.1.1.1/32, 201.1.1.2/32) with associated BFR-ids into IGP area 2.

After the path calculation, the BIER forwarding plane is built. When R1/R2 receives multicast packet which should be sent to Rm/Rn/Rx/Ry, R1/R2 encapsulates the packet with BIER header and send it into the upper area. When R3/R4 receives the packet, R3/R4 forwards the packet into IGP area 1 and area 2 according to the BIER forwarding table without doing the decapsulation and re-encapsulation actions.

Obviously, in order to build the large BIER domain, the BFR-id of edge router in each IGP area MUST NOT be overlapped.

3. Advertisement

According to [RFC8279], each BFER needs to have a unique (in each sub-domain) BFR-id, and each BFR and BFER floods itself BIER info sub-TLV and associated sub-sub-TLVs in the BIER domain. To keep consistent with the definition in [RFC8444], [I-D.ietf-bier-ospfv3-extensions], and [RFC8401], BIER info sub-TLV defined in [RFC8401] and BIER sub-TLV defined in [RFC8444], [I-D.ietf-bier-ospfv3-extensions] is reused to convey the BFR-id information. OSPF extended Prefix Opaque LSA [RFC7684] and TLVs 235, 237 defined in [RFC5120], or TLVs 135 [RFC5305], or TLV 236 [RFC5308] are still used to carry the BFR-id/ BFR-prefix information, etc.

The key parameters got from the original routing protocol should be adapted to the format of next routing protocol, such as BFR-prefix, BFR-id, subdomain-id, and so on. Some parameters like BAR, MT-ID has local significance, So they should be set to same values with BIER proxy own advertisement when BIER proxy advertise them to the next routing area.

And as the two BIER info sub-sub-TLVs (sub-TLVs) including MPLS encapsulation and BSL conversion also have local significance. The information carried in these two sub-sub-TLV need not, but MAY, be advertised to next routing area.

In the same example, when R3 advertises the prefixes of Rm and Rn into the upper area, R3 may advertise the prefixes one by one, that is R3 advertises 201.1.1.1/32 with associated BFR-id, and R3 advertises 201.1.1.2/32 with associated BFR-id. If there is dozens of edge routers in area 1, R3 advertises dozens of prefixes and associated BFR-ids into the upper area. When R3 advertises the prefixes from the upper area into area 1, R3 advertises the prefixes of R1 and R2 with associated BFR-ids separately, and R3 advertises the prefixes of Rx and Ry which come from R4 into area 1 one by one. R4 does it as well.

3.1. BIER proxy range sub-TLV

In some cases unicast default route and aggregated/ summarized routes are used in some routing areas and routers in next area can not see the specific BFR-prefix routes from original area. Like in figure 3, in case R3/R4 does not advertise specific ISIS unicast routes to OSPF area and only advertises unicast default route or aggregated/ summarized route to OSPF area 1/2, when R3/R4 advertises BIER info sub-TLV to OSPF area 1/2, R3 MUST advertise the prefix with default route or aggregated/ summarized route. In that case, multiple BFR-ids will be mapped to one prefix. In order to advertise BFR-ids optimally, we define a new BIER proxy range sub-TLV to advertise the information of BFR-ids.

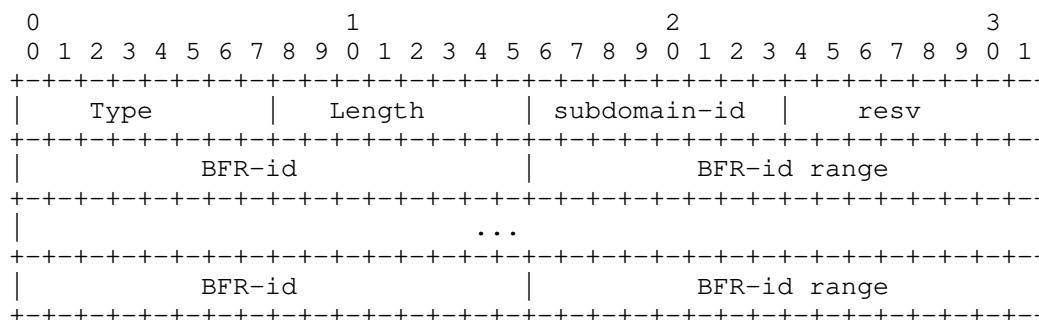


figure 4

- o Type: 8-bit unsigned integer. TBD to indicate the BIER proxy range sub-TLV.
- o Length: 8-bit unsigned integer. Length of the BIER proxy range sub-TLV in 4-octet units, not including the first 4 octets.

- o Subdomain-id: 8-bit unsigned integer. The subdomain-id from original advertisement.
- o resv: 8-bit unsigned integer. The reserved field.
- o BFR-id: 16-bit unsigned integer. The first BFR-id from original advertisement.
- o BFR-id range: 16-bit unsigned integer. The range of BFR-ids with one subdomain-id.

The BIER proxy range sub-TLV is attached to the aggregated/ summarized route prefix or default route prefix. The summarized/ aggregated/ default prefix may need multiple BIER proxy range sub-TLVs when the BFR-ids covered by the prefix are allocated from different ranges (even if they're from a single range but some BFR-ids in the range map to the BIER prefixes that are covered by a different summarized/ aggregated prefix, then that single large range needs to be broken into smaller ranges).

When a BFR receives a default/summary route with a BIER proxy range sub-TLV, it builds a BIRT route with that default/summary prefix. It also builds multiple BIFT entries for each BFR-IDs covered in the proxy range sub-TLV, using the same forwarding information as in the BIRT route. It is possible that a BFR-ID is covered in the proxy range sub-TLV of multiple default/summary routes. In that case, ECMP can be used and longest match SHOULD be used. For example, if ABR1 advertises default/summary route P1 while ABR2 advertises a more specific summary route P2 and both have a proxy range sub-TLV that covers BFR-ID 100, then the BIFT entry for BFR-ID 100 only follows the P2 route in BIRT.

The proxy range sub-TLV can also be attached to a host BIER prefix itself. Consider the situation where BGP-LU [RFC8277] is used for a seamless MPLS [I-D.ietf-mpls-seamless-mpls] environment. An ABR/ASBR advertises BIER prefixes via BGP over an area/AS to other ABR/ASBRs but the BIER prefixes are not advertised into the IGP for the area/AS. The ABR/ASBR does advertise the BIER prefix for itself into the IGP for the area/AS, with a BIER proxy range sub-TLV to cover the BFR-IDs for BFRs that the ABR/ASBR has learned (either through IGP or through BGP signaling). When an internal BFR receives it, it treats as if a default summary route had been received with the proxy range sub-TLV. Note that this imaginary default summary route is only for the purpose of building BIRT/BIFT entries and not used for unicast forwarding.

With this scheme, even though the BIER prefixes are not advertised into the IGP for the area/AS and unicast traffic for those BIER

prefixes are tunneled through, corresponding BIFT entries are maintained inside the area/AS for the purpose of efficient BIER forwarding. Otherwise, BIER forwarding through the area/AS would be tunneled just like unicast case.

The range in the BIER proxy range sub-TLV can be as granular as to advertise individual BFR-ids. Though a larger range can increase advertisement efficiency, it requires careful planning for BFR-id assignment.

When the proxy range sub-TLV is used, the mapping between a BIER prefix and its BFR-id is no longer conveyed in the routing underlay. As a result, the mapping must be provided by other means, e.g. in the multicast overlay.

3.2. Proxy range sub-TLV usage

In the same example of figure 3, in case there are 40 edge routers in area 1, the BFR-ids of area 1 start from 51 to 90, and the prefixes of these routers start from 201.1.1.1/32 to 201.1.1.40/32. These prefixes are not overlapped with the prefixes in any other area.

When R3 advertises these prefixes into the upper area, the proxy range sub-TLV can be used to optimize the advertisement. That is R3 can advertise only one prefix 201.1.1.0/24, with the BFR-id set to 51, the BFR-id range set to 40, into the upper area. Then the BIER overlay is built among R1, R2, Rm, Rn, Rx and Ry. R3 and R4 need not maintain the multicast overlay states.

When R3 advertises the prefixes from the upper area and area 2 into area 1, R3 may advertise only one default route (0.0.0.0/0) into area 1 if one or more continuous BFR-id range can be attached. Suppose that the BFR-id in the upper area starts from 1001 to 1050, the BFR-id in area 2 starts from 201 to 250, and these ranges are not overlapped with the ranges in any other area. Suppose that the sub-domain ID is 1, the BIER proxy range sub-TLV may be advertised like this:

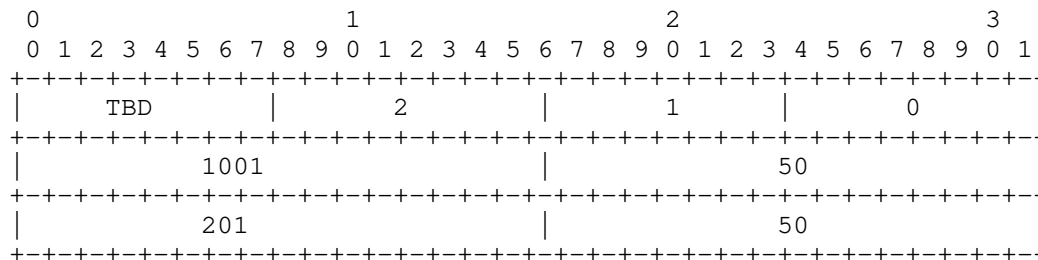


figure 5

The optimized summary/ aggregated or default prefix can be generated by the operation policy which is configured by the network administrator.

In case the range of BFR-ids in one area is overlapped with the BFR-ids in any other area, the proxy range sub-TLV can not be used. In the same example above, if the BFR-ids in area 1 are 21, 31, 41, etc., the BFR-ids in area 2 are 22, 32, 42, etc., even if the summarized prefixes are not overlapped with the prefixes in any other area, when R3 advertises the summarized prefixes in area 1 into the upper area, the proxy range sub-TLV may not optimize the advertisement.

After the forwarding plane is built, when R1 receives multicast packet, and the receivers of this flow are connected by Rm and Rx, R1 encapsulates BIER header in front of the flow packet with BFR-ids set to Rm and Rx. When R3/R4 receives the packet, R3/R4 need not decapsulate and re-encapsulate the packet. R3/R4 just forwards the packet according to the BIER forwarding table. When the packet reaches Rm and Rx, Rm and Rx remove the BIER header and forward it to receivers.

4. IANA Considerations

IANA is requested to set up a new types of sub-TLV (TLV) registry value for BIER proxy range advertisement in OSPF, ISIS, BGP, etc.

5. Security Considerations

Implementations must assure that malformed TLV and Sub-TLV permutations do not result in errors which cause hard protocol failures.

6. Acknowledgements

The authors would like to thank Stig Venaas for his valuable comments and suggestions.

7. References

7.1. Normative References

[I-D.ietf-bier-idr-extensions]

Xu, X., Chen, M., Patel, K., Wijnands, I., and T. Przygienda, "BGP Extensions for BIER", draft-ietf-bier-idr-extensions-07 (work in progress), September 2019.

- [I-D.ietf-bier-ospfv3-extensions]
Psenak, P., Nainar, N., and I. Wijnands, "OSPFv3 Extensions for BIER", draft-ietf-bier-ospfv3-extensions-02 (work in progress), May 2020.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <<https://www.rfc-editor.org/info/rfc5120>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <<https://www.rfc-editor.org/info/rfc5308>>.
- [RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <<https://www.rfc-editor.org/info/rfc7684>>.
- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.
- [RFC8401] Ginsberg, L., Ed., Przygienda, T., Aldrin, S., and Z. Zhang, "Bit Index Explicit Replication (BIER) Support via IS-IS", RFC 8401, DOI 10.17487/RFC8401, June 2018, <<https://www.rfc-editor.org/info/rfc8401>>.
- [RFC8444] Psenak, P., Ed., Kumar, N., Wijnands, IJ., Dolganow, A., Przygienda, T., Zhang, J., and S. Aldrin, "OSPFv2 Extensions for Bit Index Explicit Replication (BIER)", RFC 8444, DOI 10.17487/RFC8444, November 2018, <<https://www.rfc-editor.org/info/rfc8444>>.

7.2. Informative References

- [I-D.ietf-mpls-seamless-mpls]
Leymann, N., Decraene, B., Filsfils, C., Konstantynowicz, M., and D. Steinberg, "Seamless MPLS Architecture", draft-ietf-mpls-seamless-mpls-07 (work in progress), June 2014.

- [RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <<https://www.rfc-editor.org/info/rfc7761>>.
- [RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <<https://www.rfc-editor.org/info/rfc8277>>.

Authors' Addresses

Zheng(Sandy) Zhang
ZTE Corporation

EMail: zzhang_ietf@hotmail.com

Bo Wu
Individual

EMail: w1973941761@163.com

Zhaohui Zhang
Juniper Networks

EMail: zzhang@juniper.net

IJsbrand Wijnands
Cisco Systems, Inc.

EMail: ice@cisco.com

Yisong Liu
China Mobile

EMail: liuyisong.ietf@gmail.com

BIER
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2018

Z. Zhang
A. Przygienda
Juniper Networks
A. Dolganow
H. Bidgoli
Nokia
I. Wijnands
Cisco Systems
A. Gulko
Thomson Reuters
March 5, 2018

BIER Underlay Path Calculation Algorithm and Constraints
draft-zzhang-bier-bar-ipa-00

Abstract

This document specifies general rules for interaction between the BAR and IPA fields defined in [I-D.ietf-bier-isis-extensions] and [I-D.ietf-bier-ospf-bier-extensions].

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Terminologies	2
2. Introduction	2
3. General Rules for the BAR and IPA fields	3
3.1. When BAR Is Not Used	4
4. IANA Considerations	4
5. Acknowledgements	4
6. References	4
6.1. Normative References	4
6.2. Informative References	5
Authors' Addresses	5

1. Terminologies

Familiarity with BIER protocols and procedures is assumed. Some terminologies are listed below for convenience.

[To be added].

2. Introduction

In the BIER architecture, packets with a BIER encapsulation header are forwarded to the neighbors on the underlay paths towards the BFERs. For each sub-domain, the paths are calculated in the underlay topology for the sub-domain, following a calculation algorithm specific to the sub-domain. The <topology, algorithm> could be congruent or incongruent with unicast. The topology could be a default topology, a multi-topology [RFC5120] topology. The algorithm could be a generic IGP algorithm (e.g. SPF) or could be a BIER specific one defined in the future.

In [I-D.ietf-bier-isis-extensions] and [I-D.ietf-bier-ospf-bier-extensions], an 8-bit BAR field and 8-bit IPA field are defined to signal the BIER specific algorithm and generic IGP Algorithm respectively and only value 0 is allowed for both fields currently. This document specifies the general rules for the two fields and their interaction when either or both fields are not 0.

3. General Rules for the BAR and IPA fields

For a particular sub-domain, all routers SHOULD be provisioned with and signal the same BAR and IPA values. When a BFR discovers another BFR advertising different BAR or IPA value from its own provisioned, it MUST treat the advertising BFR as incapable of supporting BIER for the sub-domain. How incapable routers are handled is outside the scope of this document.

It is expected that both the BAR and IPA values could have both algorithm and constraints semantics. To generalize, we introduce the following terms:

- o BC: BIER-specific Constraints
- o BA: BIER-specific Algorithm
- o RC: Generic Routing Constraints
- o RA: Generic Routing Algorithm
- o BCBA: BC + BA
- o RCRA: RC + RA

A BAR value corresponds to a BCBA, and a IPA value corresponds to a RCRA. Any of the RC/BC/BA could be "NULL", which means there are no corresponding constraints or algorithm.

For a particular topology X (which could be a default topology or multit-topology topology) that a sub-domain is associated with, a router calculates the underlay paths according to its provisioned BCBA and RCRA the following way:

1. Apply the BIER constraints, resulting in BC(X).
2. Apply the routing constraints, resulting in RC(BC(X)).
3. Select the algorithm AG as following:

- A. If BA is NULL, AG is set to RA.
 - B. If BA is not NULL, AG is set to BA.
4. Run AG on RC(BC(X)).

3.1. When BAR Is Not Used

The BIER Algorithm registry established by [I-D.ietf-bier-isis-extensions] and also used in [I-D.ietf-bier-ospf-bier-extensions] has value 0 for "No BIER specific algorithm is used". That translates to NULL BA and NULL BC. Following the rules defined above, the IPA value alone identifies the calculation algorithm and constraints to be used for a particular sub-domain when BAR is 0.

4. IANA Considerations

No IANA Consideration is requested in this document.

5. Acknowledgements

The authors thanks Alia Atlas, Eric Rosen, Senthil Dhanaraj and many others for their suggestions and comments. In particular, the BCBA/RCRA representation for the interaction rules is based on Alia's write-up.

6. References

6.1. Normative References

- [I-D.ietf-bier-isis-extensions]
Ginsberg, L., Przygienda, T., Aldrin, S., and Z. Zhang,
"BIER support via ISIS", draft-ietf-bier-isis-
extensions-09 (work in progress), February 2018.
- [I-D.ietf-bier-ospf-bier-extensions]
Psenak, P., Kumar, N., Wijnands, I., Dolganow, A.,
Przygienda, T., Zhang, Z., and S. Aldrin, "OSPF Extensions
for BIER", draft-ietf-bier-ospf-bier-extensions-15 (work
in progress), February 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,
<<https://www.rfc-editor.org/info/rfc2119>>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.

6.2. Informative References

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <<https://www.rfc-editor.org/info/rfc5120>>.

Authors' Addresses

Zhaohui Zhang
Juniper Networks

EMail: zzhang@juniper.net

Antoni Przygienda
Juniper Networks

EMail: prz@juniper.net

Andrew Dolganow
Nokia

EMail: andrew.dolganow@nokia.com

Hooman Bidgoli
Nokia

EMail: hooman.bidgoli@nokia.com

IJsbrand Wijnands
Cisco Systems

EMail: ice@cisco.com

Arkadiy Gulko
Thomson Reuters

EMail: arkadiy.gulko@thomsonreuters.com

BIER
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2018

Zhaohui. Zhang
Shaowen. Ma
Juniper Networks
Zheng. Zhang
ZTE Corporation
March 5, 2018

Supporting BIER with RIFT
draft-zzhang-bier-rift-00

Abstract

This document specifies extensions to RIFT protocol to support BIER.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

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

Table of Contents

1. Terminologies	2
2. Introduction	2
3. Advertising BIER Information For non-MPLS Encapsulation . . .	3
4. Advertising BIER Information Northbound	4
5. Advertising BIER Information Southbound	4
5.1. Local BIER Information	4
5.2. Proxied BFR-ID Ranges	4
6. Information Elements Schema	5
6.1. bier.thrift	5
6.2. Additions to encoding.thrift	6
7. IANA Considerations	6
8. Acknowledgements	6
9. References	6
9.1. Normative References	6
9.2. Informative References	6
Authors' Addresses	7

1. Terminologies

Familiarity with BIER and RIFT protocols and procedures is assumed. Some terminologies are listed below for convenience.

[To be added.]

2. Introduction

BIER [RFC8279] ... (to be expanded)

RIFT [I-D.przygienda-rift] is a new protocol specifically designed for CLOS and fat-tree network topologies. As a hybrid between Link State Routing and Distance Vector Routing, it does LSR in northbound (towards the spine) and DVR in southbound (towards the leaves).

[I-D.ietf-bier-isis-extensions] and [I-D.ietf-bier-ospf-bier-extensions] specify ISIS/OSPF extensions to support BIER in an ISIS/OSPF domain. The same approach applies to RIFT in the northbound LSR.

[I-D.zwzw-bier-prefix-redistribute] specifies methods to advertise BIER information via default or summary/aggregate routes advertised

from one IGP area/domain to another. Similar approach applies to RIFT in the southbound DVR.

BIER encapsulation, whether it is based on MPLS or not, is covered in [RFC8296]. However, the OSPF/ISIS extensions for BIER only covers signaling needed for MPLS encapsulation. RIFT is targeted at DC deployments, where MPLS may not be used. This document covers signaling for both BIER MPLS and non-MPLS encapsulation with RIFT.

The details are provided in following sections.

3. Advertising BIER Information For non-MPLS Encapsulation

In the BIER architecture, a BIER sub-domain may have multiple BitStringLengths (BSLs) and multiple Encapsulations (Encaps). A single multicast packet coming from outside the BIER sub-domain may be sent as multiple BIER packets, one for each set that is identified by a SetID (SI). An incoming BIER packet is forwarded according to a BIFT for the <SD,Encap,BSL,SI> tuple. Each BIFT is identified by a 20-bit opaque number (BIFT-ID) in the packet.

With MPLS encapsulation, the BIFT-ID for an incoming BIER packet is simply an MPLS label allocated by the receiving BFR for the BIFT. For each <SD,BSL> tuple, OSPF/ISIS advertises a block of contiguous labels, one label for each SI needed for the tuple, in the MPLS Encapsulation sub-sub-TLV as part of the BIER sub-TLV, which is attached to the Extended Reachability TLV (ISIS case) or the Extended Prefix TLV (OSPF case) for the BFR's BIER Prefix.

With non-MPLS encapsulation, the BIFT-ID in the packet is at the same position as the label in MPLS encapsulation case. Its semantics is no different from the MPLS case in that as an 20-bit opaque value, it leads to the BIFT according to which the BIER packet is forwarded. Beyond the semantics, there are two differences from the MPLS case though:

- o MPLS infrastructure is not needed.
- o While each BFR could allocate local BIFT-IDs independently and advertise them just like in MPLS case, for the same <SD,Encap,BSL,SI> tuple all BFRs could optionally auto-derive or be provisioned with the same BIFT-ID and no signaling is needed in that case.

One may consider that if MPLS would allow to use consistently provisioned BIER labels on all BFRs, then the second difference listed above does not exist anymore.

In this specification, if locally significant BIFT-IDs are to be used with non-MPLS encapsulation, the BIFT-IDs are advertised the same way as in the MPLS case - by a BIFT-ID block, which is a block of contiguous labels in MPLS case or a block of contiguous opaque 20-bit values in non-MPLS case. The only difference is the type of encapsulation.

If consistently provisioned or auto-derived BIFT-IDs are used with non-MPLS encapsulation, then no BIFT-ID block is signaled. Just the encapsulation type is signaled.

4. Advertising BIER Information Northbound

Nothing special here compared to OSPF/ISIS. A node's local BIER information as described in the previous section is attached to a local BIER Prefix. Details to be added.

5. Advertising BIER Information Southbound

5.1. Local BIER Information

Similar to the northbound case, a node's local BIER information is attached to a local BIER prefix that is advertised southbound.

5.2. Proxied BFR-ID Ranges

On the southbound, a node advertises a default route, plus certain prefixes to prevent blackholing or suboptimal routing upon link failures. Those prefixes and default route are like the summary routes and default route in [I-D.zwzw-bier-prefix-redistribute], and similarly they carry BFR-IDs corresponding to the covered BIER Prefixes.

Consider a RIFT network with a BIER sub-domain of 200 BFIR/BFERS. Each non-leaf node is provisioned that BFR-ID 1-200 are used. Suppose a node X advertise southbound a default route RT1 and disaggregation routes RT2/RT3. RT2 and RT3 MUST advertise BFR-IDs covered by them (e.g. BFR-ID 100/102/150 covered by RT2 and BFR-ID 101/103 covered by RT3), while the default route RT1 can always advertise that all BFR-ID 1~200 are covered by it and does not need to exclude BFR-ID 100/102/150 and 101/103 that are covered by RT2/RT3. When a southern node receives RT1 and RT2/RT3, it installs BFR-ID 100/102/150 in its BIFT according to RT2, 101/103 in its BIFT according to RT3, and installs other BFR-IDs (or just a default route) in its BIFT according to RT1.

6. Information Elements Schema

This document introduces a bier.thrift schema with definitions to be used in RIFT encoding.thrift.

6.1. bier.thrift

```

typedef i8      SubdomainIdType
typedef i16     BfrIdType
typedef i8      BARType
typedef i8      IPAType
typedef i16     BSLType      /* Number of bits */
typedef i32     BiftIdType   /* Only the most significant 20 bits are used */
/

enum EncapsulationType {
    mpls          = 0;
    non-mpls      = 1;
}

/* Similar to the label range in OSPF/ISIS extensions for BIER */
struct BiftIdBlock {
    1: required BiftIdType      bift_id_base;
    2: required i8              bift_id_range;
}

/* Similar to the MPLS Encapsulation sub-sub-TLV in OSPF/ISIS */
struct EncapStruct {
    1: required EncapsulationType encap_type;
    2: required BSLType           bsl;
    3: optional BiftIdBlock       bift_id_block;
}

/*BIER node information. Similar to BIER sub-TLV in OSPF/ISIS. */
struct BierInfo {
    1: required SubdomainIdType  subdomain_id;
    2: required BfrIdType        bfr_id;
    3: required BARType          bar;
    4: required IPAType          ipa;
    5: required EncapStruct      encaps;      /* one or more */
}

struct ProxyBfrIdRange {
    1: required SubdomainIdType  subdomain_id;
    2: required BfrIdType        bfr_id_base;
    3: required BSLType          bfr_id_range;
}

```

6.2. Additions to encoding.thrift

The PrefixAttributes in encoding.rift now has two optional elements:

```
struct PrefixAttributes {  
    ...  
    2: optional BierInfo      bier_info;      /* BIER info for a  
                                              * local BIER Prefix */  
    3: optional ProxyBfrIdRange proxy_bfr_id; /* one or more proxy  
                                              * BFR-ID ranges covered  
                                              * by this prefix */  
}
```

7. IANA Considerations

8. Acknowledgements

9. References

9.1. Normative References

- [I-D.przygienda-rift]
Przygienda, T., Sharma, A., Atlas, A., and J. Drake,
"RIFT: Routing in Fat Trees", draft-przygienda-rift-05
(work in progress), March 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A.,
Przygienda, T., and S. Aldrin, "Multicast Using Bit Index
Explicit Replication (BIER)", RFC 8279,
DOI 10.17487/RFC8279, November 2017,
<<https://www.rfc-editor.org/info/rfc8279>>.
- [RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A.,
Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation
for Bit Index Explicit Replication (BIER) in MPLS and Non-
MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January
2018, <<https://www.rfc-editor.org/info/rfc8296>>.

9.2. Informative References

[I-D.ietf-bier-isis-extensions]

Ginsberg, L., Przygienda, T., Aldrin, S., and Z. Zhang,
"BIER support via ISIS", draft-ietf-bier-isis-
extensions-09 (work in progress), February 2018.

[I-D.ietf-bier-ospf-bier-extensions]

Psenak, P., Kumar, N., Wijnands, I., Dolganow, A.,
Przygienda, T., Zhang, Z., and S. Aldrin, "OSPF Extensions
for BIER", draft-ietf-bier-ospf-bier-extensions-15 (work
in progress), February 2018.

[I-D.zwzw-bier-prefix-redistribute]

Zhang, Z., Bo, W., Zhang, Z., and I. Wijnands, "BIER
Prefix Redistribute", draft-zwzw-bier-prefix-
redistribute-00 (work in progress), January 2018.

Authors' Addresses

Zhaohui Zhang
Juniper Networks

EMail: z Zhang@juniper.net

Shaowen Ma
Juniper Networks

EMail: mashao@juniper.net

Zheng Zhang
ZTE Corporation

EMail: zhang.zheng@zte.com.cn