Network Working Group
Internet-Draft
Expires: November 30, 2004

A. Doria (co-editor)
ForCES Protocol Design Team

June 2004

ForCES Protocol Specification
draft-doria-forces-protocol-01.txt

Status of this Memo

Copyright Notice

Abstract

This is the first draft of the ForCES protocol specification produced by the protocol design team. It is presented to the WG for consideration as a WG work item.

Authors

The participants in the ForCES Protocol Team, authors and co-editors, of this draft, are:
Ligang Dong, Zhejiang Gongshang University

Avri Doria, ETRI
Ram Gopal, Nokia
Robert Haas, IBM
Jamal Hadi Salim, Znyx
Hormuzd M Khosravi, Intel
Weiming Wang, Zhejiang Gongshang University

Acknowledgment

The authors of this draft would like to acknowledge and thank all the protocol proposal authors including Alex Audu, Steven Blake, Chaoping Wu, Jeff Pickering, Yunfei Guo, and Guangming Wang for their contributions. We would also like to thank David Putzolu, and Patrick Droz for their comments and suggestions on the protocol.

Table of Contents

1. Introduction

The protocol defined in this ID is the first draft of the effort being made by the ForCES Protocol Design team to develop a ForCES protocol that meets the requirements set out in [RFC3654].

The team was created by the WG chairs of the ForCES group in consultation with the Routing Area ADs after the authors of the various qualifying drafts, i.e. drafts submitted by the deadline which met the essential requirements outline in [RFC3654], reached consensus on being able to develop a common protocol proposal.

While this draft has rough consensus within the design team, there are still issues that need to be resolved. These issues will continue to be discussed within the protocol design team and hopefully, on the general ForCES WG email list.

The protocol design team was defined by the WG chairs to be composed of seven participants includes two representatives from each of the 3 protocol proposals and 1 participant who had not contributed to the proposed protocols.

The ForCES protocol defined in this draft is being presented to the ForCES WG for consideration as a WG item corresponding to the charter item:

> Specification of IP-based protocol for transport of the controlled objects. When the control and forwarding devices are separated beyond a single hop, ForCES will make use of an existing RFC2914 compliant L4 protocol with adequate reliability, security and congestion control (e.g. TCP, SCTP) for transport purposes.

1.1 Sections of this document

Section 2 provides a glossary of terminology used in the specification.

Section 3 provides an overview of the protocol including a discussion on the protocol framework, descriptions of the protocol layer (PL) and a transport mapping layer (TML), as well as of the ForCES protocol mechanisms.

While this document does not define the TML, Section 4 details the services that the TML must provide.

The Forces protocol is defined to have a common header for all other message types. The header is defined in Section 5, while the protocol messages are defined in Section 6.

Section 8 describes mechanism in the protocol to support high availability mechanisms including redundancy and fail over. Section 9 defines the security mechanisms provided by the PL and TML.

2. Definitions

This document follows the terminologies defined by the ForCES Requirements in [RFC3654] and by the ForCES framework in [RFC3746]. This document also uses the terminologies defined by ForCES FE model in [FE-MODEL]. We copy the definitions of some terminologies as below:

Addressable Entity (AE) - A physical device that is directly addressable given some interconnect technology. For example, on IP networks, it is a device to which we can communicate using an IP address; and on a switch fabric, it is a device to which we can communicate using a switch fabric port number.

Forwarding Element (FE) - A logical entity that implements the ForCES protocol. FEs use the underlying hardware to provide per-packet processing and handling as directed/controlled by a CE via the ForCES protocol.

Control Element (CE) - A logical entity that implements the ForCES protocol and uses it to instruct one or more FEs how to process packets. CEs handle functionality such as the execution of control and signaling protocols.

Pre-association Phase - The period of time during which a FE Manager (see below) and a CE Manager (see below) are determining which FE and CE should be part of the same network element.

Post-association Phase - The period of time during which a FE does know which CE is to control it and vice versa, including the time during which the CE and FE are establishing communication with one another.

FE Model - A model that describes the logical processing functions of a FE.

FE Manager (FEM) - A logical entity that operates in the pre-association phase and is responsible for determining to which CE(s) a FE should communicate. This process is called CE discovery and may involve the FE manager learning the capabilities of available CEs. A FE manager may use anything from a static configuration to a pre-association phase protocol (see below) to determine which CE(s) to use. Being a logical entity, a FE manager might be physically combined with any of the other logical entities such as FEs.

CE Manager (CEM) - A logical entity that operates in the pre-association phase and is responsible for determining to which FE(s) a CE should communicate. This process is called FE discovery

and may involve the CE manager learning the capabilities of available FEs. A CE manager may use anything from a static configuration to a pre-association phase protocol (see below) to determine which FE to use. Being a logical entity, a CE manager might be physically combined with any of the other logical entities such as CEs.

ForCES Network Element (NE) - An entity composed of one or more CEs and one or more FEs. To entities outside a NE, the NE represents a single point of management. Similarly, a NE usually hides its internal organization from external entities.

High Touch Capability - This term will be used to apply to the capabilities found in some forwarders to take action on the contents or headers of a packet based on content other than what is found in the IP header. Examples of these capabilities include NAT-PT, firewall, and L7 content recognition.

Datapath -- A conceptual path taken by packets within the forwarding plane inside an FE.

LFB (Logical Function Block) type -- A template representing a fine-grained, logically separable and well-defined packet processing operation in the datapath. LFB types are the basic building blocks of the FE model.

LFB (Logical Function Block) Instance -- As a packet flows through an FE along a datapath, it flows through one or multiple LFB instances, with each implementing an instance of a certain LFB type. There may be multiple instances of the same LFB in an FE's datapath. Note that we often refer to LFBs without distinguishing between LFB type and LFB instance when we believe the implied reference is obvious for the given context.

LFB Metadata -- Metadata is used to communicate per-packet state from one LFB to another, but is not sent across the network. The FE model defines how such metadata is identified, produced and consumed by the LFBs, but not how metadata is encoded within an implementation.

LFB Attribute -- Operational parameters of the LFBs that must be visible to the CEs are conceptualized in the FE model as the LFB attributes. The LFB attributes include, for example, flags, single parameter arguments, complex arguments, and tables that the CE can read or/and write via the ForCES protocol (see below).

LFB Topology -- Representation of how the LFB instances are logically interconnected and placed along the datapath within one FE. Sometimes it is also called intra-FE topology, to be distinguished from inter-FE topology.

FE Topology -- A representation of how the multiple FEs within a single NE are interconnected. Sometimes this is called inter-FE topology, to be distinguished from intra-FE topology (i.e., LFB topology).

Inter-FE Topology -- See FE Topology.

Intra-FE Topology -- See LFB Topology.

Following terminologies are defined by this document:

ForCES Protocol - While there may be multiple protocols used within the overall ForCES architecture, the term "ForCES protocol" refers only to the protocol used at the Fp reference point in the ForCES Framework in RFC3746 [RFC3746]. This protocol does not apply to CE-to-CE communication, FE-to-FE communication, or to communication between FE and CE managers. Basically, the ForCES protocol works in a master-slave mode in which FEs are slaves and CEs are masters. This document defines the specifications for this ForCES protocol.

ForCES Protocol Layer (ForCES PL) -- A layer in ForCES protocol architecture that defines the ForCES protocol messages, the protocol state transfer scheme, as well as the ForCES protocol architecture itself (including requirements of ForCES TML (see below)). Specifications of ForCES PL are defined by this document.

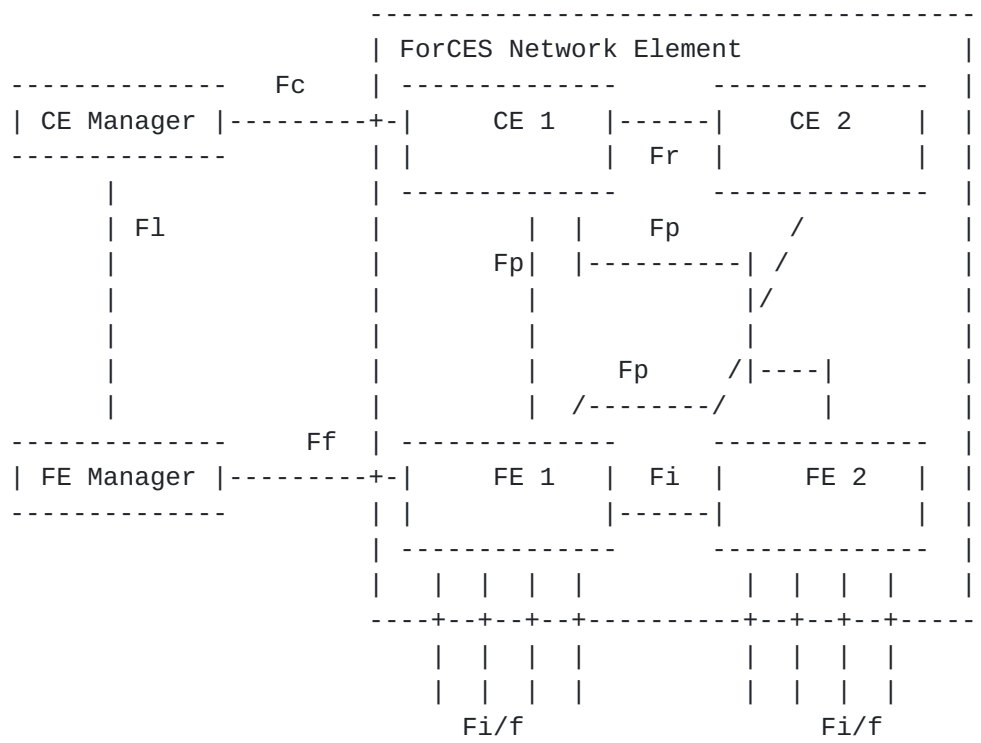
ForCES Protocol Transport Mapping Layer (ForCES TML) -- A layer in ForCES protocol architecture that specifically addresses the protocol message transportation issues, such as how the protocol messages are mapped to different transport media (like TCP, IP, ATM, Ethernet, etc), and how to achieve and implement reliability, multicast, ordering, etc. The ForCES TML is specifically addressed in a separate ForCES TML Specification document.

3. Overview

The reader is referred to the Framework document [RFC3746], and in particular sections 3 and 4, for architectural overview and where and how the ForCES protocol fits in. There may be some content overlap between the framework document and this section in order to provide clarity.

3.1 Protocol Framework

Figure 1 below is reproduced from the Framework document for clarity. It shows a NE with two CEs and two FEs.



Fp: CE-FE interface
Fi: FE-FE interface
Fr: CE-CE interface
Fc: Interface between the CE Manager and a CE
Ff: Interface between the FE Manager and an FE
Fl: Interface between the CE Manager and the FE Manager
Fi/f: FE external interface

Figure 1: ForCES Architectural Diagram

The ForCES protocol domain is found in the Fp Reference Point. The Protocol Element configuration reference points, Fc and Ff also play a role in the booting up of the Forces Protocol. The protocol

element configuration is out of scope of the ForCES protocol but is touched on in this document since it is an integral part of the protocol pre-association phase.

Figure 2 below shows further breakdown of the Fp interface by example of a MPLS QoS enabled Network Element.

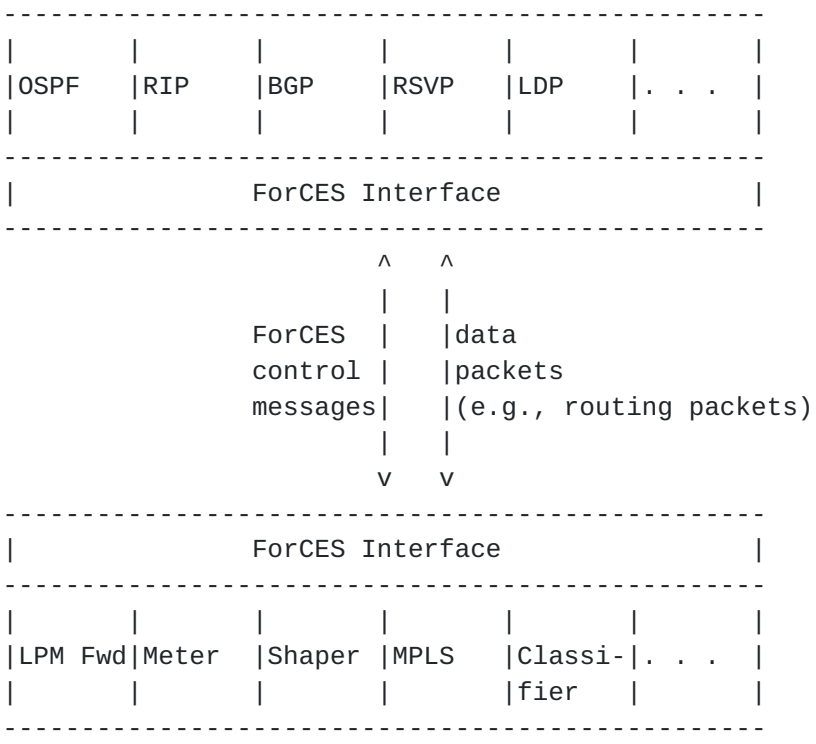


Figure 2: Examples of CE and FE functions

The ForCES Interface shown in Figure B constitutes two pieces: the PL and TML layer.

This is depicted in Figure 3 below.

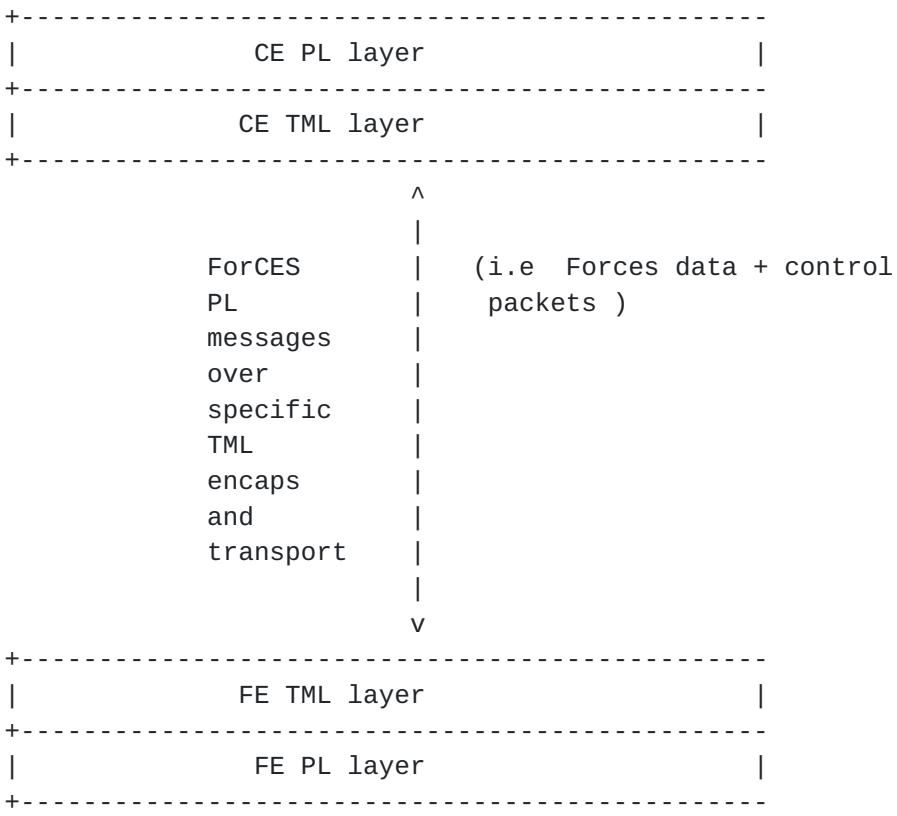


Figure 3: ForCES Interface

The PL layer is in fact the ForCES protocol. Its semantics and message layout are defined in this document. The TML Layer is necessary to connect two ForCES PL layers as shown in Figure 3 above. The TML is out of scope for this document but is within scope of ForCES. This document defines requirements the PL needs the TML to meet.

Both the PL and TML layers are standardized by the IETF. While only one PL layer is defined, different TMLs are expected to be standardized. To interoperate the TML layer at the CE and FE are expected to be of the same definition.

On transmit, the PL layer delivers its messages to the TML layer. The TML layer delivers the message to the destination TML layer(s). On receive, the TML delivers the message to its destination PL layer(s).

3.1.1 The PL layer

The PL is common to all implementations of ForCES and is standardized

by the IETF as defined in this document. The PL layer is responsible for associating an FE or CE to an NE. It is also responsible for tearing down such associations. An FE uses the PL layer to throw various subscribed-to events to the CE PL layer as well as respond to various status requests issued from the CE PL. The CE configures both the FE and associated LFBs attributes using the PL layer. In addition the CE may send various requests to the FE to activate or deactivate it, reconfigure its HA parametrization, subscribe to specific events etc. More details in section Section 6.

3.1.2 The TML layer

The TML layer is essentially responsible for transport of the PL layer messages. The TML is where the issues of how to achieve transport level reliability, congestion control, multicast, ordering, etc are handled. It is expected more than one TML will be standardized. The different TMLs each could implement things differently based on capabilities of underlying media and transport. However, since each TML is standardized, interoperability is guaranteed as long as both endpoints support the same TML. All ForCES Protocol Layer implementations should be portable across all TMLs, because all TMLs have the same top edge semantics as defined in this document.

3.1.3 The FEM/CEM Interface

The FEM and CEM components, although valuable in the setup and configurations of both the PL and TML layers, are out of scope of the ForCES protocol. The best way to think of them are as configurations/parameterizations for the PL and TML before they become active (or even at runtime based on implementation). In the simplest case, the FE or CE read a static configuration file which they use as the FEM/CEM interface. RFC 3746 has a lot more detailed descriptions on how the FEM and CEM could be used. We discuss the pre-association phase where the CEM and FEM play briefly in section Section 3.2.1.

An example of typical things FEM/CEM would configure would be TML specific parameterizations such as:

a. how the TML connection should happen (example what IP addresses to use, transport modes etc);
b. the ID for the FE or CE would also be issued at this point.
c. Security parameterization such as keys etc.
d. Connection association parameters

Example "send up to 3 association messages each 1 second apart" Vs " send up to 4 association messages with increasing exponential timeout".

3.2 ForCES Protocol Phases

ForCES in relation to NEs involves two phases: the Pre-Association phase where configuration/initialization/bootup of the TML and PL layer happens, and the association phase where the ForCES protocol operates.

3.2.1 Pre-association

The ForCES interface is configured during the Pre association phase. In a simple setup, the configuration is static and is read from some saved config file. All the parameters for the association phase are well known after the pre association phase is complete. A protocol such as DHCP may be used to retrieve the config parameters instead of reading from a static config file. Note this will still be considered static pre-association. Dynamic configuration may also happen in the Fc, Ff and Fl reference points. Vendors may use their own proprietary service discovery protocol to pass the parameters.

We reproduce some scenarios from the Framework Document to show a pre-association example.

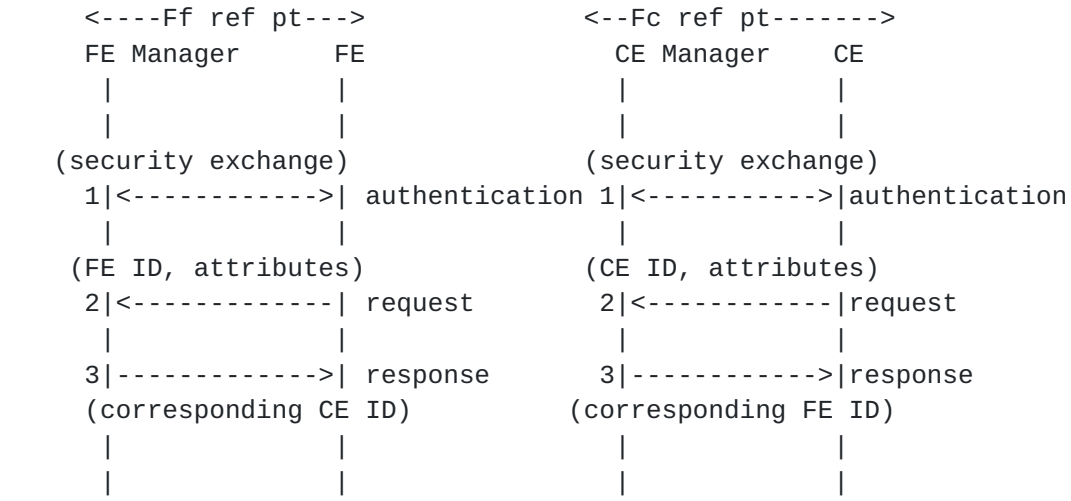


Figure 4: Examples of a message exchange over the Ff and Fc reference points

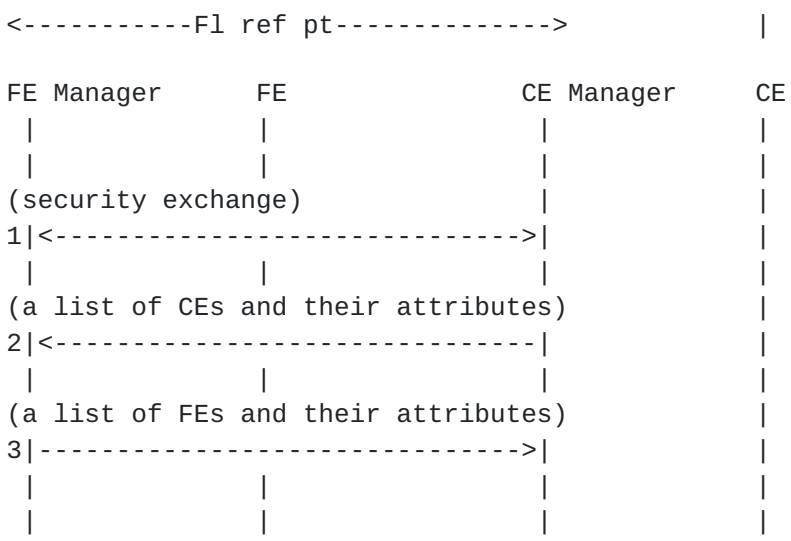


Figure 5: An example of a message exchange over the Fl reference point

Before the transition to the association phase, the FEM will have established contact with the appropriate CEM component. Initialization of the ForCES interface will be completed, and Authentication and capabilities discovery may be complete as well. Both the FE and CE would know how to connect to each other for configuration, accounting, identification and authentication purposes. Both sides are also knowledgeable of all necessary protocol parameters such as timers, etc. The Fl reference point may continue to operate during the association phase and may be used to force a disassociation of an FE or CE. Because the Pre-association phase is out of scope, we do not discuss these details any further. The reader is referred to the framework document for more detailed discussion.

3.2.2 Post-association

In this phase, the FE and CE components talk to each other using the ForCES protocol (PL over TML) as defined in this document. There are three sub-phases: Association setup state, Established State, and Association teardown state.

3.2.2.1 Association setup state

The FE attempts to join the NE. The FE may be rejected or accepted. Once granted access into the NE, capabilities exchange happen with the CE querying the FE. Armed with the FE capability knowledge, the CE can offer an initial configure (possibly to restore state) and query certain attributes within either an LFB or the FE itself.

A lot more details are provided in the protocol scenarios section.

On successful completion of this state, the FE joins the NE and is moved to the Established State.

3.2.2.2 Association Established state

In this state the FE is continuously updated or queried. The FE may also send asynchronous event notifications to the CE or synchronous heartbeat notifications. This continues until a termination is initiated by either the CE or FE.

Refer to section on protocol scenarios for more details.

3.3 Protocol Mechanisms

Various semantics are exposed to the protocol users via the PL header including: Transaction capabilities, atomicity of transactions, two phase commits, batching/parallelization, High Availability and failover as well as command windows.

3.3.1 Of Transactions, Atomicity and 2 Phase Commits

A transaction is a sequence of operations that is guaranteed to be atomic in the presence of any failures by the CEs or FEs. Operation in this sense implies the PL operation within the message body TLV. An example of a transaction could be a config PL msg with a sequence of operations: "route-add A B,C:route-del X" (each operation in its own TLV).

If a transaction is split across more than one message, then all message batch must arrive at the destination before they are executed on either the LFB or FE. All operations are executed serially in the order specified by the transaction. If any of the sequence of operations in a transaction fails then the transaction is declared as a failure. This is an all-or-nothing approach and is needed to ensure consistency of a transaction across multiple FEs.

A transaction may be atomic within an FE alone or across the NE. In both cases the atomic requirement for a transaction MUST be met. The PL message header exposes ability to mark a start of transaction and end of transaction using flags. These flags can be used to derive a classical transactional two phase commit[ACID paper ref here].

3.3.2 FE Protocol Object

The FE Protocol Object is a logical entity in each FE that is used to control the ForCES protocol. It is defined in the same fashion as

LFBs. The FE Protocol Object can be manipulated using the standard Config/Query messages. The FE Protocol Object Type ID is assigned the value 0x1. The FE Protocol Instance ID is assigned the value 0x1. There must always be one and only one instance of the FE Protocol Object in an FE. The values of the attributes in the FE Protocol Object have pre-defined default values that are specified here. Unless explicit changes are made to these values using Config messages from the CE, these default values MUST be used for the operation of the protocol.

The FE Protocol Object consists of the following elements:

- FE Protocol events that can be subscribed/unsubscribed:
  - FE heartbeat
  - FE TML events (TBD)
- FE Protocol capabilities:
  - Supported ForCES protocol version(s) by the FE
  - Supported ForCES FE model(s) by the FE
  - Some TML capability description(s)
- FE Protocol attributes:
  - current version of the ForCES protocol
  - current version of the FE model
  - Association Expiry Timer
  - Heartbeat Interval
  - Primary CE
  - FE failover and restart policy

The FE Object (referred to as "FE attributes" in the model draft) should not be confused with the FE Protocol Object. The FE Object contains attributes relative to the FE itself, and not to the operation of the ForCES protocol between the CE and the FE. Such attributes can be FEState (refer to model draft), vendor, etc. The FE Object Type ID is assigned the value 0x2. The FE Protocol Instance ID is assigned the value 0x1. There must always be one and only one instance of the FE Object in an FE.

The FE Object consists of the following elements

- FE Events:
  - FEStatusChange (FE Up/Down/Active/Inactive/Failover)
  - FE DoS alert
  - FE capability change
- FE attributes:
  - FE Behavior Exp. Timer
  - HA Mode

3.3.3 Scaling by Concurrency

It is desirable that the PL layer is not the bottleneck when larger bandwidth pipes become available. To pick a mythical example in

todays terms, if a 100Gbps pipe was made available and there is sufficient work then the PL layer should take advantage and use all of the 100Gbps pipe. Two semantics are allowed to achieve this. The first one is batching and the second one is a command window. Batching is ability to send multiple commands (such as config) in one PDU. The size of the batch will be affected by amongst other things the path MTU. The commands may be part of the same transaction or unrelated transactions which are independent of each other. Command windowing allows for pipelining of independent transactions which do not affect each other. Each independent transaction could be one or more batches.

3.3.3.1 Batching

TBD

3.3.3.2 Command Pipelining

TBD

4. TML Requirements

The requirements below are expected to be delivered by the TML. This text does not define how such mechanisms are delivered. As an example they could be defined to be delivered via hardware or inter-TML protocol level schemes.

Each TML must describe how it contributes to achieving the listed ForCES requirements. If for any reason a TML does not provide a service listed below a justification needs to be provided.

1. Reliability
   As defined by RFC 3654, section 6 #6.
2. Security
   TML provides security services to the ForCES PL. TML layer should support following security services and describe how they are achieved.
   * Endpoint authentication of FE and CE.
   * Message Authentication
   * Confidentiality service
3. Congestion Control
   The congestion control scheme used needs to be defined. Additionally, under what circumstances is notification sent to the PL to notify it of congestion.
4. Uni/multi/broadcast addressing/delivery if any
   If theres any mapping between PL and TML level Uni/Multi/ Broadcast addressing it needs to be defined.
5. Timeliness

   Editorial Note: Does the TML allow for obsoleting msgs? If yes, it needs to say how.
6. HA decisions
   It is expected that availability of transport links is the TMLs responsibility. However, on config basis, the PL layer may wish to participate in link failover schemes and therefore the TML must provide this capability.
   Please refer to the HA Section Section 8 for details.
7. Encapsulations used.
   Different types of TMLs will encapsulate the PL messages on different types of headers. The TML needs to specify the encapsulation used.
8. Prioritization
   It is expected that the TML will be able to handle up to 8 priority levels needed by the PL layer and will provide preferential treatment.
   TML needs to define how this is achieved.

4.1 TML Parameterization

It is expected that it should be possible to use a configuration reference point, such as the FEM or the CEM, to configure the TML.

Some of the parameters may include:

- PL ID
- Connection Type and associated data. For example if a TML uses IP/TCP/UDP then parameters such as TCP and UDP ports, IP addresses need to be configured.
- number of transport connections
- Connection Capability, such as bandwidth, etc.
- Allowed/Supported Connection QoS policy (or Congestion Control Policy)

5. Common Header

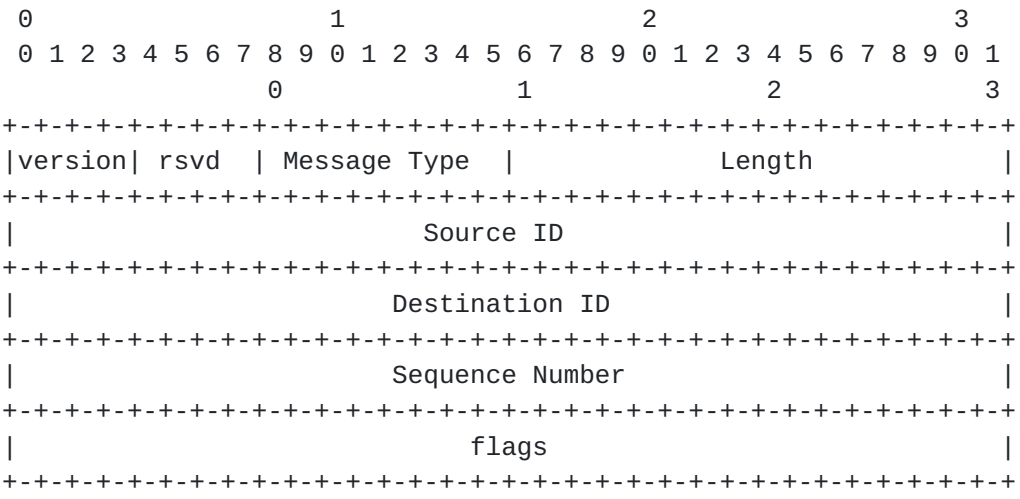


Figure 6: Common Header

The message is 32 bit aligned.

Version (4 bit):

Version with a 2 bit major and 2 bit minor.

rsvd (4 bit):

Unused at this point. A receiver should not interpret this field.

Command (8 bits):

Commands are defined in Section 6.

Source ID (32 bit):

Dest ID (32 bit):

* Above are 32 bit IDs which recognize the termination point. Ideas discussed so far are desire to recognize if ID belongs to FE or CE by inspection. Suggestions for achieving this involves partitioning of the ID allocation. Another alternative maybe to use flags to indicate direction (this avoids partition).
* IDs will allow multi/broad/unicast
* Addressing
  a. As ForCES may run between multiple CEs and FEs and over different protocols such as IPv4 and IPv6, or directly over Ethernet or other switching-fabric interconnects, it is necessary to create an addressing scheme for ForCES entities. Mappings to the underlying TML-level addressing can then be defined as appropriate.
  b. Fundamentally, unique IDs are assigned to CEs and FEs. A split address space is used to distinguish FEs from CEs. Even though we can assume that in a large NE there are typically two or more orders of magnitude more FEs than CEs, the address space is split uniformly for simplicity.

c. Special IDs are reserved for FE broadcast, CE broadcast, and NE broadcast.

d. Subgroups of FEs belonging, for instance, to the same VPN, may be assigned a multicast ID. Likewise, subgroups of CEs that act, for instance, in a back-up mode may be assigned a multicast ID. These FEs and CE multicast IDs are chosen in a distinct portion of the ID address space. Such a multicast ID may comprise FEs, CEs, or a mix of both.

e. As a result, the address space allows up to 2^30 (over a billion) CEs and the same amount of FEs.

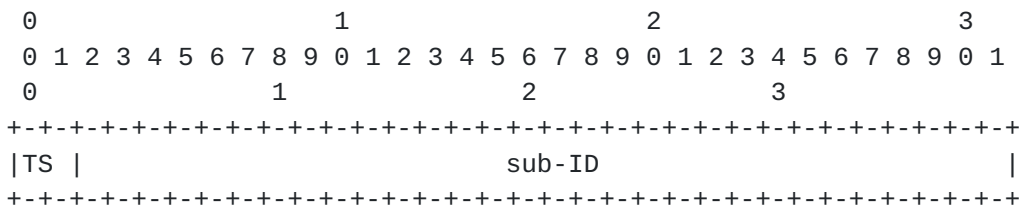


Figure 7: ForCES ID Format

f. The ForCES ID is 32 bits. The 2 most significant bits called Type Switch (TS) are used to split the ID space as follows:

TS	Corresponding ID range	Assignment
0b00	0x00000000 to 0x3FFFFFFF	FE IDs (2^30)
0b01	0x40000000 to 0x7FFFFFFF	CE IDs (2^30)
0b10	0x80000000 to 0xBFFFFFFF	reserved
0b11	0xC0000000 to 0xFFFFFFEF	multicast IDs (2^30 - 16)
0b11	0xFFFFFFF0 to 0xFFFFFFFC	reserved
0b11	0xFFFFFFFD	all CEs broadcast
0b11	0xFFFFFFFE	all FEs broadcast
0b11	0xFFFFFFFF	all FEs and CEs (NE) broadcast

g. It is desirable to address multicast and/or broadcast messages to some LFB instances of a given class. For instance, assume FEs FEa and FEb:

- FEa has LFBs LFBaX1 and LFBaX2 of class X
- similarly, FEb has two LFBs LFBbX1 and LFBbX2 of class X.

A broadcast message should be addressable to only LFBs LFBaX1 and LFBbX1 (this can be the case for instance if these two LFBs belong to the same VPN). To achieve this, a VPN ID (3 octets OUI and 4 octets VPN Index) as defined in RFC 2685 should be used within the ForCES message body as a TLV.

As an alternative, a particular multicast ID MAY be associated to a given VPN ID through some configuration means. Messages delivered to such a multicast ID MUST only be applied to LFBs belonging to that VPN ID.

Sequence (32 bits)
Unique to a PDU. [Discussion: There may be impact on the effect of subsequence numbers].

length (16 bits):
length of header + the rest of the message in DWORDS (4 byte increments).

Flags(32 bits):
Identified so far:

* ACK indicator(2 bit)
  The description for using the two bits is:
  'NoACK' (00) | 'SuccessACK'(01) | 'UnsuccessACK'(10) | 'ACKAll' (11)
* unknown/undecided.
* Throttle flag?
* batch (2 bits)
* Atomicity (1 or 2 bits)

Editorial Note: There are several open issues, listed below, in the header which still need to be settled:

1. Parallelization of PL Windowing/subsequence Someone to look into ISCSI
2. events and replies and relation to peer to peer vs master slave
3. We need to discuss whether some of the Flags such as those for Atomicity, Batching are needed in the common header or only belong to the Config message.

6. Protocol Messages

6.1 Association Messages

The ForCES Association messages are used to establish and teardown associations between FEs and CEs.

6.1.1 Association Setup Message

This message is sent by the FE to the CE to setup a ForCES association between them. This message could also be used by CEs to join a ForCES NE, however CE-to-CE communication is not covered by this protocol.

Message transfer direction:
FE to CE
Message Header:
The Message Type in the header is set MessageType= 'Association Setup'. The ACK flag in the header is ignored, because the setup message will always expect to get a response from the message receiver (CE) whether the setup is successful or not. The Src ID (FE ID) may be set to O in the header which means that the FE would like the CE to assign a FE ID for the FE in the setup response message.
Message body:
The setup message body consists of one optional TLV, the format of which is as follows:

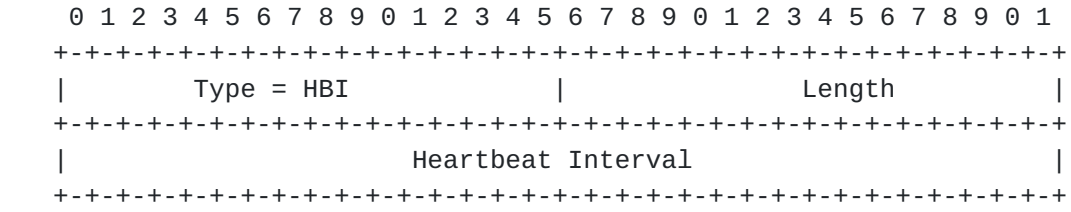


Figure 8

Type (16 bits):
Currently only one Type defined, HBI-heart beat interval.
Length (16 bits):
Length of the TLV including the T and L fields, in bytes.
Heartbeat Interval (32 bits):
This indicates the current HB interval on the FE in milliseconds. A default value for this will be defined.

Editorial Note: Whether HBI belongs to the setup is still under discussion.

Editorial Note: In certain situations (such as use of multicast IDs), it might not be possible to make use of the procedure described above for the FE to dynamically obtain an ID from the CE. Such situations need to be identified.

6.1.2 Association Setup Response Message

This message is sent by the CE to the FE in response to the Setup message. It indicates to the FE whether the setup is successful or not, i.e. whether an association is established.

Message transfer direction:
CE to FE
Message Header:
The Message Type in the header is set MessageType= 'Setup Response'. The ACK flag in the header is always ignored, because the setup response message will never expect to get any more response from the message receiver (FE). The Dst ID in the header will be set to some FE ID value assigned by the CE if the FE had requested that in the setup message (by SrcID = 0).
Message body:
The setup response message body consists of one TLV, the format of which is as follows:

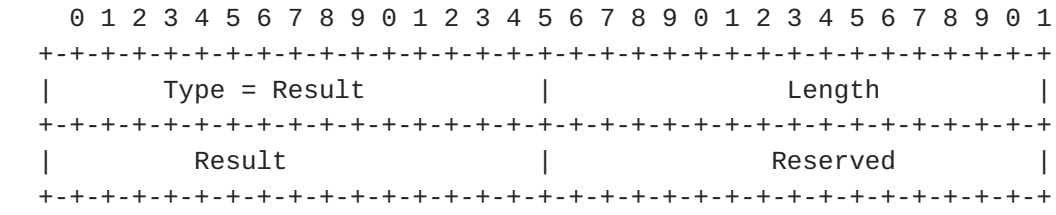


Figure 9

Type (16 bits):
Currently only one Type i.e. Setup Result is required. Other TLVs are optional.
Length (16 bits):
Length of the TLV including the T and L fields, in bytes.
Result (16 bits):
This indicates whether the setup msg was successful or whether the FE request was rejected by the CE.

6.1.3 Association Teardown Message

This message can be sent by the FE or CE to any ForCES element to end its ForCES association with that element.

Message transfer direction:
CE to FE, or FE to CE (or CE to CE)
Message Header:
The Message Type in the header is set MessageType= "Asso. Teardown"(TBD?). The ACK flag in the header is always ignored, because the teardown message will never expect to get any response from the message receiver.
Message body:
The association teardown message body consists of one TLV, the format of which is as follows:

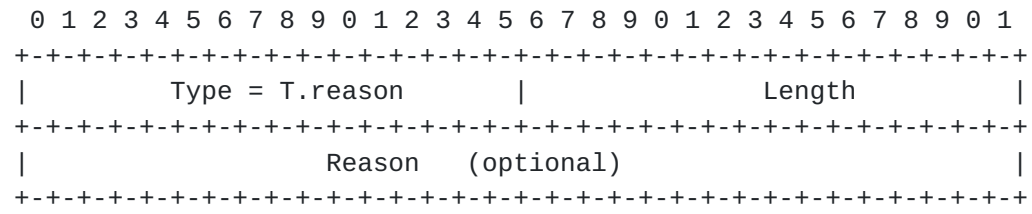


Figure 10

Type (16 bits):
Currently only one Type defined - Teardown Reason.
Length (16 bits):
Length of the TLV including the T and L fields, in bytes.
T.reason (32 bits):
This indicates the reason why the association is being terminated.

6.2 Query and Response Messages

Editorial Note: If the approach of using an FE Protocol & FE Object is fully adopted and no other reason for having FE TLVs is identified, then no distinction will be further made in the TLV types between FE* and LFB*. As a result, the Type ID and Instance ID in the TLV will also be used to identify the FE Protocol Object, with specific values as mentioned in Section 3.3.2

The ForCES query and response messages are used for one ForCES element (CE or FE) to query other ForCES element(s) for various kinds

of information. Current version of ForCES protocol limits the use of the messages only for CE to query information of FE. The information to be queried in FE can be categorized into two types:

1) FE (coarse layer) information

This type of information is about the property of an FE, taking the FE as a whole, e.g., the total available memory space in the FE. To query this type of information, we should take the whole FE as the addressing destination. Information of this type includes:

- Intra-FE topology
- Inter-FE topology, that is, LFB topology
- FE capabilities
- FE attributes
- FE statistics

Another way to recognize FE coarse layer property is to define two objects, the 'FE Protocol Object' and the 'FE Object' at this coarse layer. See Section 3.3.2 for more details.

2) LFB information

This type of information is about property of an LFB inside an FE, e.g., routing rules of a Forwarder LFB in an FE. To query this type of information, we should take the LFB as the addressing destination. Information of this type include:

- LFB capabilities
- LFB attributes
- LFB statistics

6.2.1 Query Message

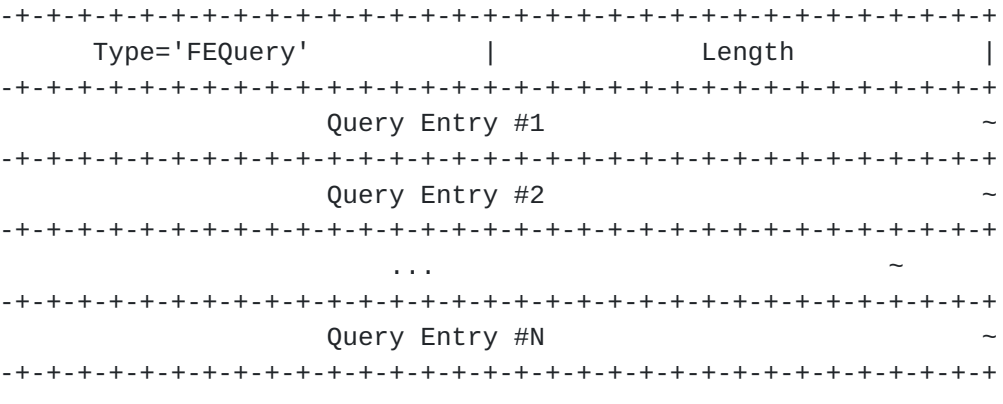
As usual, a query message is composed of a common header and a message body that consists of one or more TLV data format. Detailed description of the message is as below.

Message transfer direction:
    Current version limits the query message transfer direction only from CE to FE.
Message Header:
    The Message Type in the header is set to MessageType= 'Query'. The ACK flag in the header SHOULD be set 'ACKAll', meaning a full response for a query message is always expected. If the ACK flag is set other values, the meaning of the flag will then be ignored, and a full response will still be returned by message receiver.

Message body:

The query message body consists of (at least) one or more than one TLVs that describe entries to be queried. According to the queried information being FE (coarse layer) information or LFB information as described above, the message body TLV has different data format as below:

To query FE (coarse layer) information, the message body TLV data format is:



To query LFB information, the message body TLV data format is:

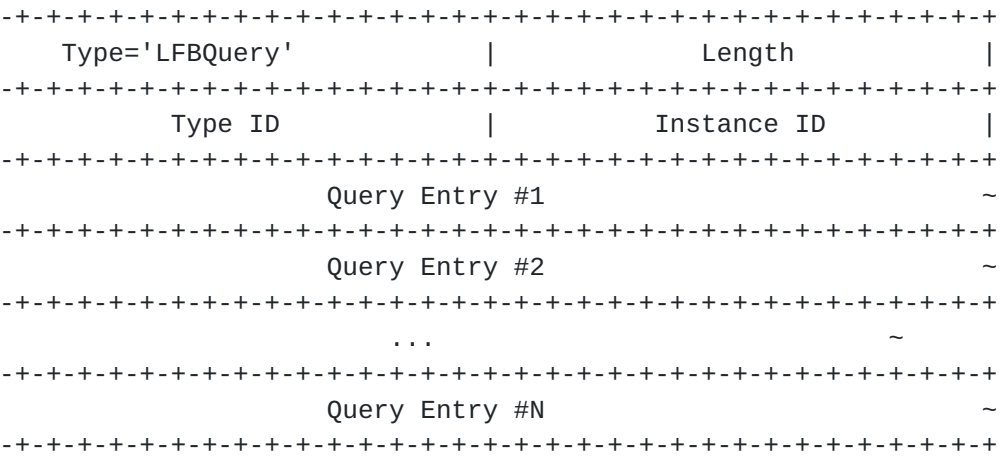


Figure 11

Editorial Note:

1. Under discussion is, when an 'FE Protocol Object' idea is adopted, above two kind of data formats may be mapped into one format, with the Type ID and Instance ID changed to Managed Object (MO) Type ID and MO Instance ID, and with two specific MO Type ID and

Instance ID for 'FE Protocol Object' and 'FE Object', and others for 'FE LFB Object'.

2. Under discussion is, do we need to support multiple objects addressing at the LFB Type and LFB Instance layer? One simple way to support multiple LFB types or instances is to use TLVs to identify the group of Type IDs and Instance IDs, rather than only one Type and Instance ID. A range of Instance IDs may also be supported in this way.

Query Entry:

This is a TLV that describes the entry to be queried, as follows:

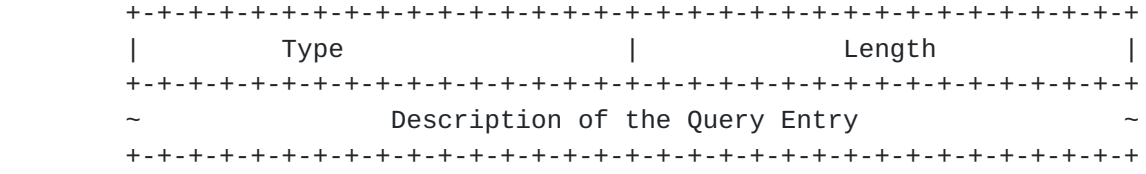


Figure 12

Type:

[Under discussion and TBD]

Editorial Note: There is a debate on how the Type here can be used. One possible use for the Type is to specify the encoding type for the TLV value. The possible encoding types are that like XML, binary ID based TLV coding, etc, therefore, the possible value for the Type may be 'XMLEncoding', 'ID-BasedBinaryEncoding', etc. The Cons say that it may be impractical to directly use XML encoding in the protocol format, leaving no encoding type other than ID-based binary encoding to be specified, hence no need to specify encoding type. Another possible use of the Type is to number the entries, by which a more flexible response based on the number may become be achieved.

Description of the Query Entry:

This field presents the detailed description about the entry to be queried. The encoding of the description is based on the ForCES FE model if the entry is defined by FE model, or based on vendor specifications if the entry is defined by vendors. Note that the encoding is responsible for the 32 bits alignment of the description field. Usually, the description should include

information about the name (or the name ID) of the entry to be queried. Occasionally, it may also include some more information, like some conditions that the queried entry is required to meet. For instance, consider a case in which a CE is going to query a Forwarder LFB in an FE for its current routing table information. In this case, CE may be interested in knowing (querying) all routing rules in the table, CE may also be interested in only knowing (querying) a few routing rules that meet some specific conditions, e.g., routing rules whose source IP address should match '210.33.X.X'. In the former case, only the name of the LFB attribute (as 'routing table') should be told in the query entry description, while in the latter case, the matching conditions should also be told in the description. Taking another policer LFB as one more example, the policer LFB may have attribute like the provisions (rules) for the policer, therefore, in the query entry TLV, we may set the queied entry name as the 'Provision'. To only set the name of the attribute as 'Provision' will dump all provision items in the LFB; while to set the attribute name and followed by some conditions will dump the provision items that meet the conditions. The index of the provision items can also be as the conditions, e.g., to set the conditions as 'the index of the provision items should range from 11 to 15', then will return query result that only include the provision items ranging from 11 to 15.

6.2.2 Query Response Message

When receiving a query message, the receiver should process the message and come up with a query result. The receiver sends the query result by use of the Query Response Message back to the query message sender. The query result can be the information being queried if the query operation is successful, or can also be error codes if the query operation fails, indicating the reasons for the failure.

A query response message is also composed of a common header and a message body consists of one or more TLVs describing the query result. Detailed description of the message is as below.

Message transfer direction:
Current version limits the query response message transfer direction only from FE to CE.
Message Header:
The Message Type in the header is set to MessageType= 'QueryResponse'. The ACK flag in the header SHOULD be set 'NoACK', meaning no further response for a query response message is expected. If the ACK flag is set other values, the meaning of the flag will then be ignored. The Sequence Number in the header

SHOULD keep the same as that of the query message to be responded, so that the query message sender can keep track of the responses.

Message body:

The message body for a query response message consists of (at least) one or more than one TLVs that describe query results to individual queried entries. According to the queried information being FE (coarse layer) information or LFB information as described above, the response message body TLV has different data format as below:

To respond to the query for FE (coarse layer) information, the response TLV has following data format:

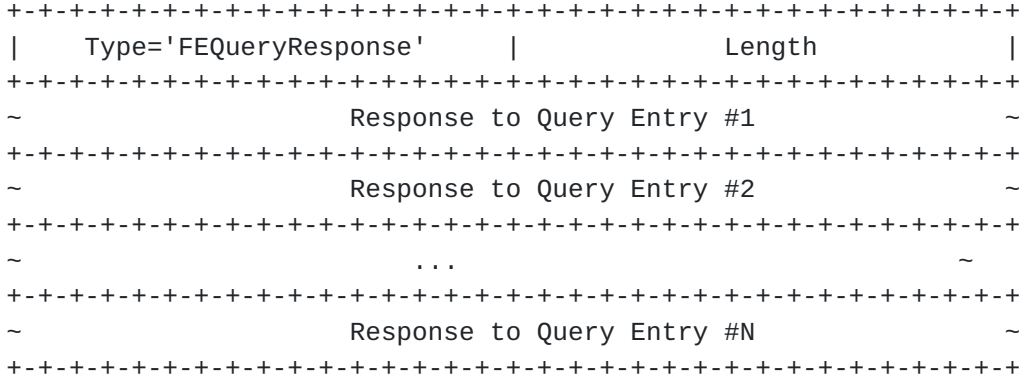


Figure 13

To respond to the query for LFB information, the response TLV has data format as:

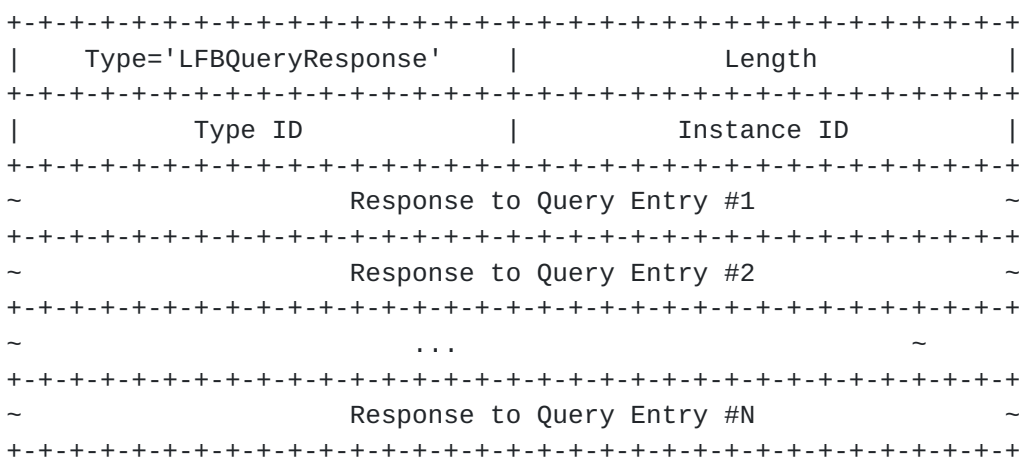


Figure 14

Response to Query Entry:
This is a TLV that describes the response to the queried entry, as follows:

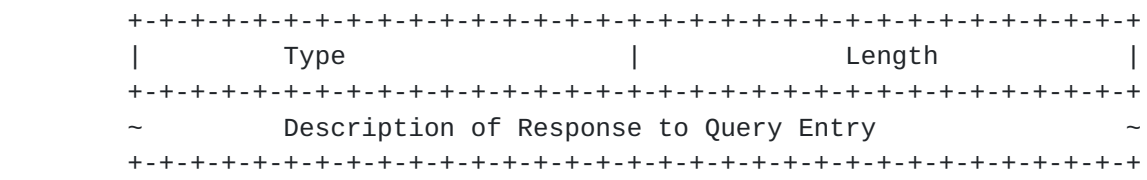


Figure 15

Type:
[Under discussion and TBD]
Description of Response to Query Entry:
This field presents the detailed description about the query result of an entry. The encoding of the description is based on the ForCES FE model if the entry is defined by FE model, or based on vendor specifications if the entry is defined by vendors. Note that the encoding is responsible for the 32 bits alignment of the description field. When the query is successful, the response result will include the information being queried. When the query is failed, the response result will usually include the information about the reason for the failure.

6.3 Configuration Messages

Editorial Note: If the approach of using an FE Protocol & FE Object is fully adopted and no other reason for having FE TLVs is identified, then no distinction will be further made in the TLV types between FE* and LFB*. As a result, the Type ID and Instance ID in the TLV will also be used to identify the FE Protocol Object, with specific values as mentioned in Section 3.3.2

The ForCES Configuration messages are used by the CEs to configure the FEs in a ForCES NE and report the results back to the CE.

6.3.1 Config Message

This message is sent by the CE to the FE to configure FE or LFB attributes. This message is also used by the CE to subscribe/ unsubscribe to FE, LFB events.

Message transfer direction:
CE to FE
Message Header:
The Message Type in the header is set MessageType= 'Config'. The ACK flag in the header is can be used by the CE to turn off any response from the FE. The default behavior is to turn on the ACK to get the config response from the FE.
Message body:
The Config message body consists of one or more TLVs, the format of a single (LFB) TLV is as follows:

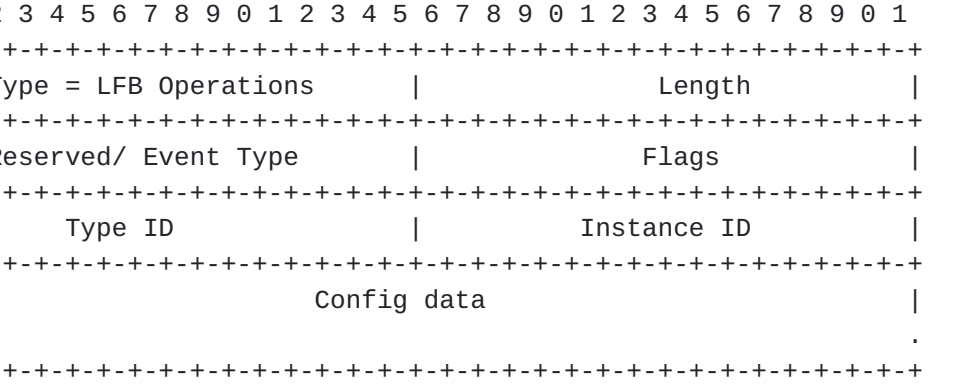


Figure 16

The format for a FE attributes TLV is as follows

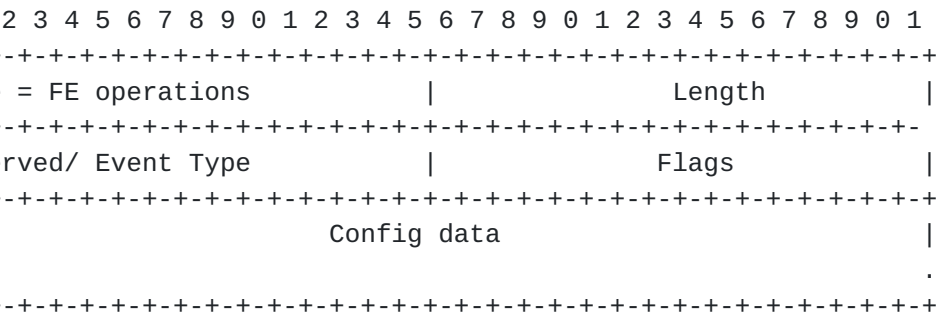


Figure 17

Type (16 bits):
This is a combination of FE, LFB attributes with operations. The operations include, ADD, DEL, UPDATE/REPLACE, DEL ALL, SUBSCRIBE, UNSUBSCRIBE, CANCEL. The following Types are defined for this TLV:

* FE Add, Del, Update, Del All, Cancel, Subscribe, Unsubscribe events
* LFB Add, Del, Update, Del All, Cancel, Subscribe, Unsubscribe events

For any of the FE attribute Types, the Type, and Instance ID fields are not present in this TLV.

Length (16 bits):
Length of the TLV including the T and L fields, in bytes.

Flags (16 bits):
These can be used to indicate Atomicity, Batching, etc.

Type ID (16 bits):
This field uniquely recognizes the LFB type.

Instance ID (16 bits):
This field uniquely identifies the LFB instance.

Config Data (variable length):
This will carry LFB specific data (single or Array LFB specific entries). The config data might itself be of the form of a TLV.

Event Type (16 bits):
For SUBSCRIBE, UNSUBSCRIBE Events Type TLVs, an Event Type field will define the Events of interest. Examples of Event Type include, All Events, FE Events, LFB Events, Packets, Packet Mirroring.

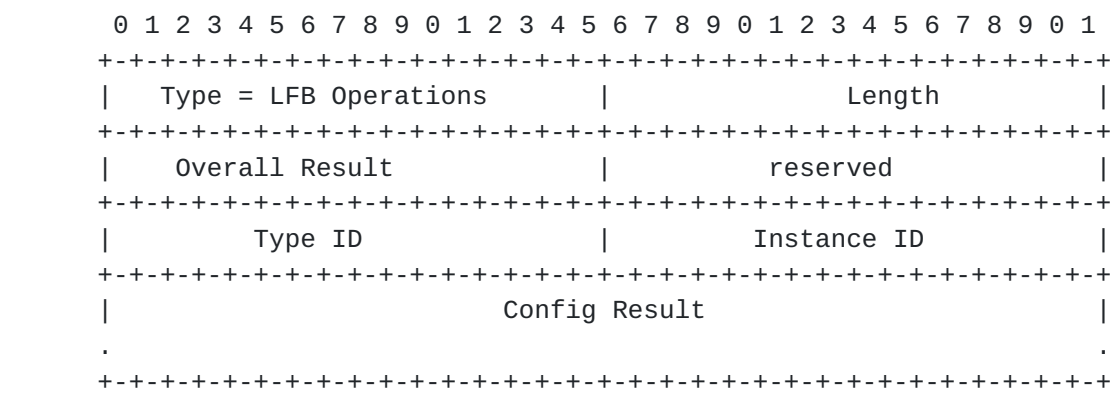
6.3.2 Config Response Message

This message is sent by the FE to the CE in response to the Config message. It indicates whether the Config was successful or not on the FE and also gives a detailed response regarding the configuration result of each attribute.

Message transfer direction:
FE to CE

Message Header:
The Message Type in the header is set MessageType= 'Config Response'. The ACK flag in the header is always ignored, because the config response message will never expect to get any more response from the message receiver (CE).

Message body:
The Config response message body consists of one or more TLVs, the format of a single TLV is as follows:



The format for a FE response TLV is as follows

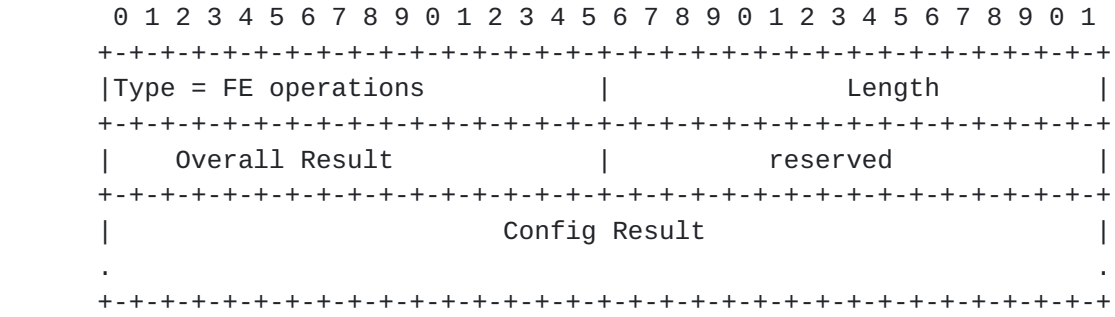


Figure 18

Type (16 bits):
Same as that for Config message. For any of the FE attribute Types, the Type, and Instance ID fields are not present in this TLV.

Length (16 bits):
Length of the TLV including the T and L fields, in bytes.

Overall Result (16 bits):
This indicates the overall result of the config message, whether it was successful or it failed.

Type ID (16 bits):
This field uniquely recognizes the LFB type.

Instance ID (16 bits):
This field uniquely identifies the LFB instance.

Config Result (variable length):
This will carry LFB specific results (single or Array LFB specific result entries). The config result might itself be of the form of a TLV.

6.4 Event Notification and Response Messages

Editorial Note: If the approach of using an FE Protocol & FE Object is fully adopted and no other reason for having FE TLVs is identified, then no distinction will be further made in the TLV types between FE* and LFB*. As a result, the Type ID and Instance ID in the TLV will also be used to identify the FE Protocol Object, with specific values as mentioned in Section 3.3.2

The Event Notification Message is used for one ForCES element to asynchronously notify one or more other ForCES elements in the same ForCES NE on just happened events in it. The Event Notification Response Message is used for the receiver of the Event Notification Message to acknowledge the reception of the event notification.

Events in current ForCES protocol can be categorized into following three types:

- Events happened in CE
- Events happened in FE at the FE coarse layer (in FE protocol object and FE object)
- Events happened in LFB inside an FE

Events can also be categorized into two classes according to whether they need subscription or not. An event in one ForCES element that needs to be subscribed will send notifications to other ForCES elements only when the other elements have subscribed to the element for the event notification. How to subscribe/unsubscribe for an event is described in the Configure Message in Section 6.3. An event that needs not to be subscribed will always send notifications to other ForCES elements when the event happens. An event definition made by ForCES FE model or by vendors will state if the event needs subscription or not.

Editorial Note: There is an argument that it is preferable to have all events subscribable.

6.4.1 Event Notification Message

As usual, an Event Notification Message is composed of a common header and a message body that consists of one or more TLV data format. Detailed description of the message is as below.

Message Transfer Direction:

FE to CE, or CE to FE

Message Header:

The Message Type in the message header is set to MessageType = 'EventNotification'. The ACK flag in the header can be set as: ACK flag ='NoACK'|'SuccessAck'|'UnsuccessACK'|'ACKAll'.

Note that the 'Success' here only means the receiver of the message has successfully received the message.

Message Body:

The message body for an event notification message consists of (at least) one or more than one TLVs that describe the notified events.

According to the different event types described above, the message body TLV has different data format, which is defined as follows:

For events generated by CE or by FE coarse layer, the TLV has following data format:

For events generated by LFB in FE, the TLV has data format as:

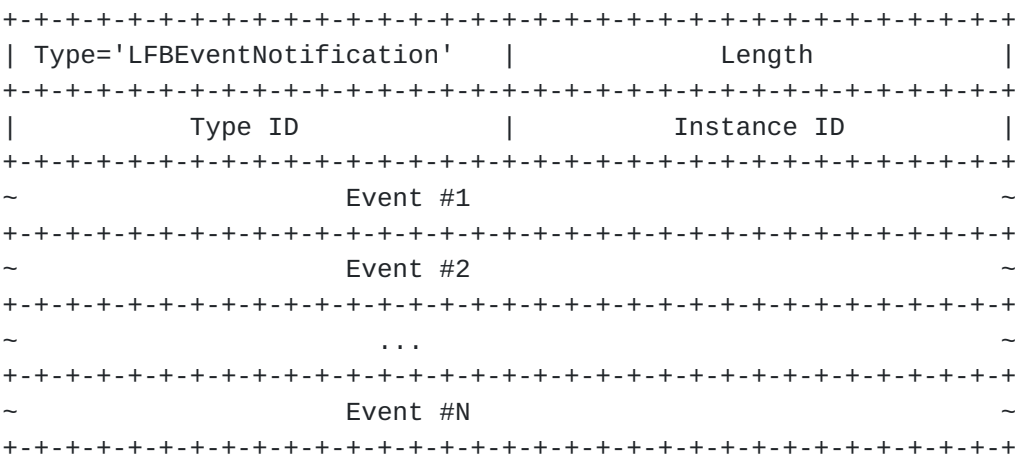


Figure 19

Event:
This is a TLV that describes the event to be notified, as follows:

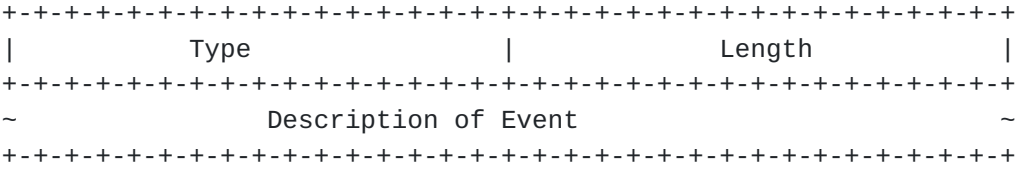


Figure 20

Type:
[TBD]
Description of Event:
This field will make a detailed description of the happened event. The encoding of the description is based on the ForCES FE model if the event is defined by FE model, or based on vendor specifications if the event is defined by vendors. Note that the encoding is responsible for the 32 bits alignment of the description field.The description will usually include the name (or the name ID) of the event. It may also include some other information like parameters that are related to the happened event.

6.4.2 Event Notification Response Message

After sending out an Event Notification Message, the sender may be interested in ensuring that the message has been received by receivers, especially when the sender thinks the event notification is vital for system management. An Event Notification Response Message is used for this purpose. The ACK flag in the Event Notification Message header are used to signal if such acknowledge is requested or not by the sender.

Detailed description of the message is as below:
Message Transfer Direction:
From FE to CE or from CE to FE, just inverse to the direction of the Event Notification Message that it responses.
Message Header:
The Message Type in the header is set MessageType= 'EventNotificationResponse'. The ACK flag in the header SHOULD be set 'NoACK', meaning no further response for the message is expected. If the ACK flag is set other values, the meaning of the flag will then be ignored. The Sequence Number in the header SHOULD keep the same as that of the message to be responded, so that the event notificatin message sender can keep track of the responses.

This contains a TLV that describe the response result as below:

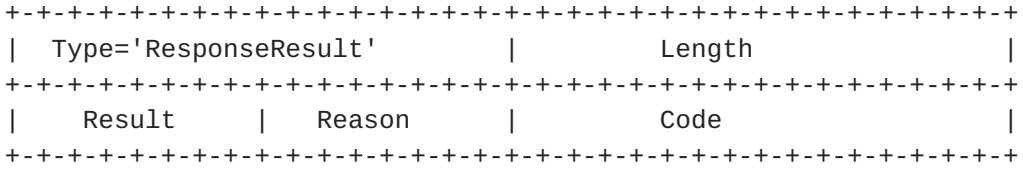


Figure 21

Result:
This describes the reception result of the event notification message as below:

Result Value	Meaning
'Success'	The event has been successfully received.
'Unsuccess'	The event has not been successfully received.

Reason, Code:
This describes the reason and possible error code when the message is not successfully received. Note that only the failure at the protocol layer rather than the transport layer can be allocated here, that is, if even the header part of the message to be responded can not be correctly received, the response to the message will not be able to be generated by the receiver.

Editorial Note: There is a debate on whether the Event Notification Response Message is necessary or not. The pro for it is some event notification senders may be interested in knowing if receivers have had success/unsuccess receptions of the events or not. An alternative to generate such response is for the protocol to define a universal ACK message so that it can act as responses for any types of messages as well as the event notification messages, when the message senders are interested in knowing whether the messages have been successfully received or not (different from the responses for the message processing results).

6.5 Packet Redirect Message

Packet redirect message is used to transfer data packets between CE and FE. Usually these data packets are IP packets, though they may sometimes associated with some metadata generated by other LFBs in the model, or they may occasionally be other protocol packets, which usually happen when CE and FE are jointly implementing some

high-touch operations. Packets redirected from FE to CE are the data packets that come from forwarding plane, and usually are the data packets that need high-touch operations in CE. Packets redirected from CE to FE are the data packets that are generated by CE and are decided by CE to put into forwarding plane in FE.

By properly configuring related LFBs in FE, a packet can also be mirrored to CE instead of purely redirected to CE, i.e., the packet is duplicated and one is redirected to CE and the other continues its way in the LFB topology.

Editorial Note: There are also discussions on how LFBs in FE model that are related to packet redirect operations should be defined. Although it is out of the scope of forces protocol, how to define the LFBs affect the Packet Redirect Message described here. Because currently it is still in progress in FE model on how to define such LFBs, we try to post some thoughts on this here for discussion. They will be removed later along with the progress of the FE model work.

Thought 1: To define LFBs called 'RedirectSink' and 'RedirectTap' for packet redirect.
An LFB in FE called 'RedirectSink' is responsible to collect data packets that need to be redirected to CE. From the perspective of the FE LFB topology, the 'RedirectSink' LFB is an LFB with only one input port and without any output port, and the input port can then be connected to any other LFB in FE model by means of a datapath in the forwarding plane. From the perspective of the ForCES protocol layer, the 'RedirectSink' LFB will generate the Packet Redirect Messages when it receives data packets from forwarding plane.

An LFB in FE called 'RedirectTap' is responsible to receive data packets that are redirected from CE. From the perspective of the FE LFB topology, the 'RedirectTap' LFB is an LFB with only one output port and without any input port, and the output port can then be connected to any other LFB in FE model by means of a datapath in the forwarding plane. From the perspective of ForCES protocol layer, the 'RedirectTap' LFB can receive the Packet Redirect Messages from CE, and un-encapsulate the data packets from the message and put them to datapaths in the forwarding plane. Actually the 'RecirectTap' LFB acts more like a transcoder that transfers the

ForCES protocol messages to normal data packets in IP forwarding plane. As a result, if we need to have redirected packets connected to some LFB (say a Scheduler) in FE model, we only need to connect the 'RedirectTap LFB to the Scheduler LFB directly via a datapath as follows:

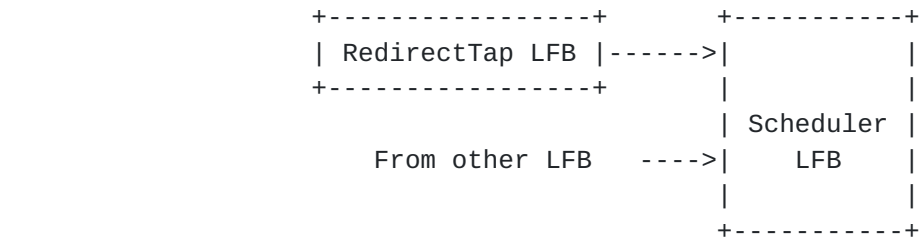


Figure 23

By use of several 'RedirectSink' LFBs and several 'RedirectTap' LFBs that connect to several different datapaths in FE forwarding plane, multiple packet redirect paths between CE and FE can be constructed.

Thought 2: There might be another way a packet could be redirected: directly by a forwarding path, e.g., by FPGA/ASIC/NP microcode. In such a case we do not need to put in a lot of smartness. Probably a link layer or even network level header is enough. The receiver demuxes it only based on some protocol type in the link layer or network transport layer. The pros for this appraoch is it may provide a fast and cost-effective path for packet redirect. The cons for this is it may more or less confuses the Fp reference point definition in ForCES framework.

We describe the Packet Redirect Message data format in details as follows:

Message Direction:

CE to FE or FE to CE

Message Header:

The Message Type in the header is set to MessageType= 'PacketRedirect'. The ACK flags in the header SHOULD be set 'NoACK', meaning no response is expected by this message. If the ACK flag is set other values, the meanings will be ignored.

Message Body:
Consists of one or more TLVs, with every TLV having the following data format:

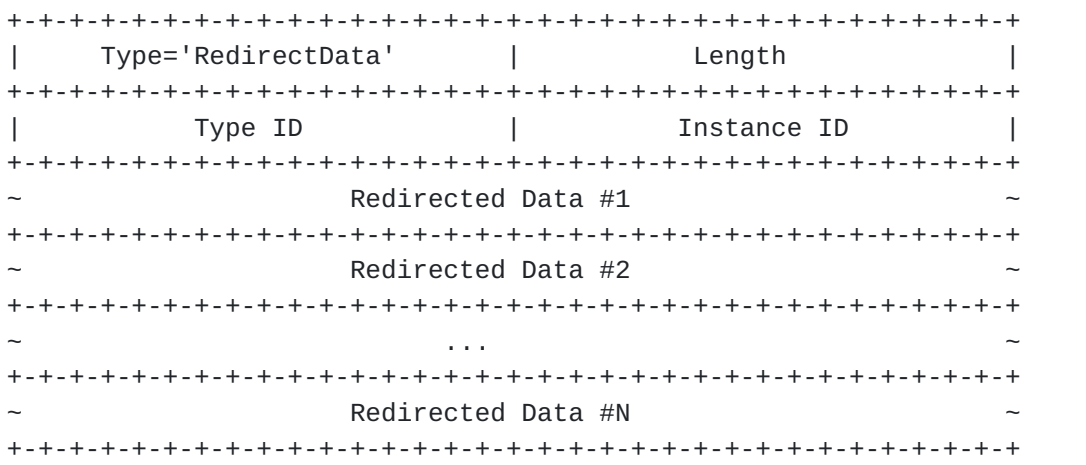


Figure 24

Type ID:
There are only two possible LFB types here, the 'RedirectSink' LFB or the 'RedirectTap' LFB. If the message is from FE to CE, the LFB type should be 'RedirectSink'. If the message is from CE to FE, the LFB type should be 'RedirectTap'.

Instance ID:
Instance ID for the 'RedirectSink' LFB or 'RedirectTap' LFB.

Redirected Data:
This is a TLV describing one packet of data to be directed via the specified LFB above. The order of the data number is also the order the data packet arrives the redirector LFB, that is, the Redirected Data #1 should arrive earlier than the Redirected Data #2 in this redirector LFB. The TLV format is as follows:

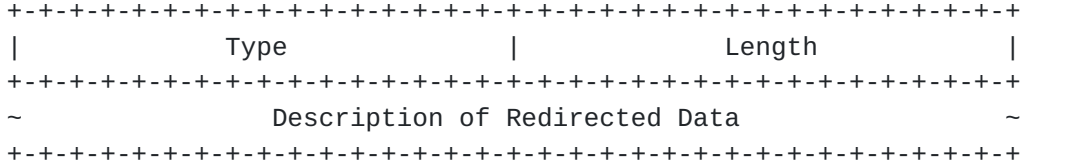


Figure 25

Type:
[TBD]

Description of Redirected Data:
This field will make a detailed description of the data to be redirected as well as the data itself. The encoding of the description is based on the ForCES FE model if the redirector LFB

is defined by FE model, or based on vendor specifications if the redirector LFB is defined by vendors. The description will usually include the name (or the name ID) of the redirected packet data (such as 'IPv4 Packet', 'IPv6 Packet'), and the packet data itself. It may also include some metadata (metadata name (or name ID) and its value)associated with the redirected data packet.

6.6 State Maintenance Messages

The State Maintenance Messages are used by the CE to change state related information on the FE.

Editorial Note: As work progresses in defining the FE model, it may happen that the messages defined here (State Maintenance messages) become redundant. For instance, FE activation/deactivation may be performed by configuring the FE State attribute in the FE Object. Such inconsistencies will be resolved

6.6.1 State Maintenance Message

This message is sent by the CE to change the state of the FE, e.g. to Activate/Deactivate the FE, shutdown the FE, etc.
Message transfer direction:
CE to FE
Message Header:
The Message Type in the header is set MessageType= 'State Maintenance'. The ACK flag in the header is can be used by the CE to turn off any response from the FE. The default behavior is to turn on the ACK to get the state maintenance response from the FE.
Message body:
The state maintenance message body consists of one or more TLVs, the format of a single TLV is as follows:

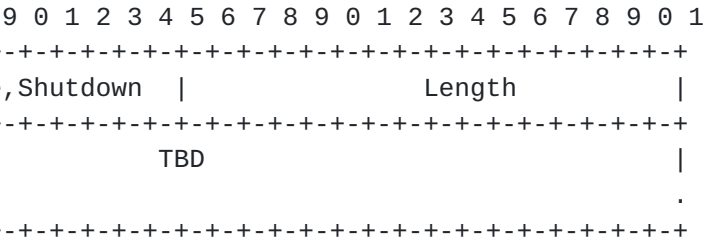


Figure 26

Type (16 bits):
These can be FE Activate, FE Deactivate, Shutdown FE. Activating an FE means asking it to forward packets, Deactivate means the FE stops forwarding packets. The default state of the FE is deactivated till it explicitly activated by the CE.
Editorial Note: These Types may be extended to include LFB Activate/Deactivate as well. However this is still being discussed.
Length (16 bits):
Length of the TLV including the T and L fields, in bytes.
FE State object (variable):
This is an TLV which can be defined and extended to represent FE specific state information. It will contain information such as the HA Mode, Primary CE ID, etc for the FE.

6.6.2 State Maintenance Response Message

This message is sent by the FE to the CE in response to the state m. message. It indicates whether the state m. was successful or not on the FE.
Message transfer direction:
FE to CE
Message Header:
The Message Type in the header is set MessageType= 'state m. Response'. The ACK flag in the header is always ignored, because the state m. response message will never expect to get any more response from the message receiver (CE).
Message body:
The state maintenance response message body consists of one or more TLVs, the format of a single TLV is as follows:

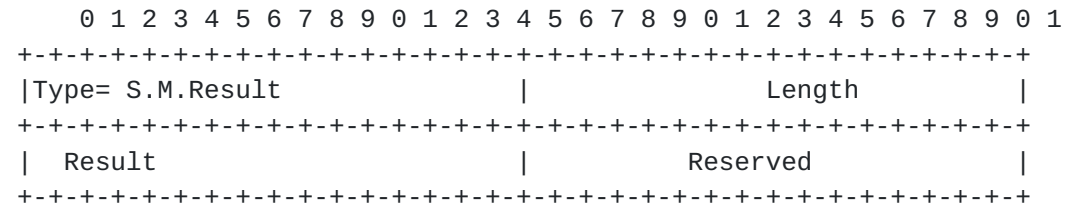


Figure 27

Type (16 bits):
Same as that for state maintenance message.
Length (16 bits):
Length of the TLV including the T and L fields, in bytes.

Overall Result (16 bits):
    This indicates the overall result of the state maintenance message, whether it was successful or it failed.

6.7 Heartbeat Message

The Heartbeat (HB) Message is used for one ForCES element (FE or CE) to asynchronously notify one or more other ForCES elements in the same ForCES NE on its liveness.

A Heartbeat Message is sent by a ForCES element periodically. The time interval to send the message is set by the Association Setup Message described in Section 6.1.1. A little different from other protocol messages, a Heartbeat messge is only composed of a common header, withe the message body left empty. Detailed description of the message is as below.

Message Transfer Direction:
    FE to CE, or CE to FE

Message Header:
    The Message Type in the message header is set to MessageType = 'Heartbeat'. The ACK flag in the header SHOULD be set to 'NoACK', meaning no response from receiver(s) is expected by the message sender. Other values of the ACK flag will always be ignored by the message receiver.

Message Body:
    The message body is empty for the Heartbeat Message, so as to grasp more efficiency for message transportation and processing.

7. Protocol Scenarios

7.1 Association Setup state

The associations among CEs and FEs are initiated via Association setup message from the FE. If a setup request is granted by the CE, a successful setup response message is sent to the FE. If CEs and FEs are operating in an insecure environment then the security association have to be established between them before any association messages can be exchanged. The TML will take care of establishing any security associations.

This is followed by capability query, topology query. When the FE is ready to start forwarding data traffic, it sends a FE UP Event message to the CE. The CE responds with a FE ACTIVATE State Maintenance message to ask the FE to go active and start forwarding data traffic. At this point the association establishment is complete. These sequences of messages are illustrated in the Figure below.

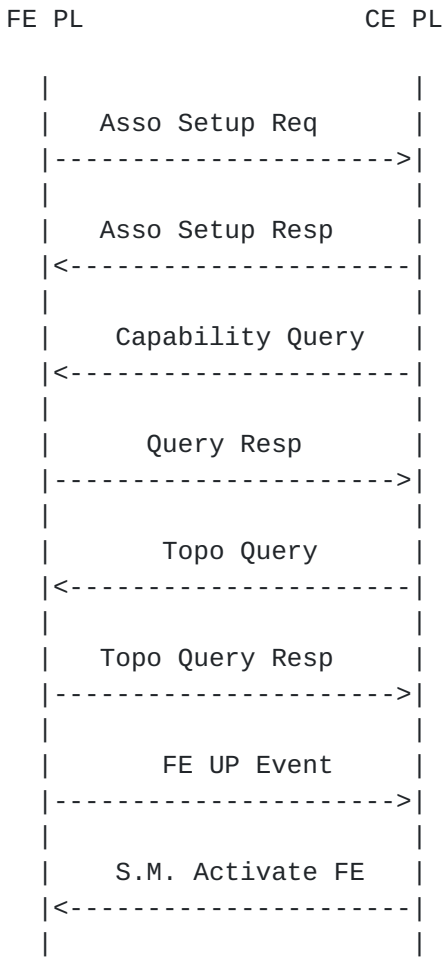


Figure 28: Message exchange between CE and FE to establish an NE association

On successful completion of this state, the FE joins the NE and is moved to the Established State or Steady state.

7.2 Association Established state or Steady State

In this state the FE is continously updated or queried. The FE may also send asynchronous event notifications to the CE or synchronous heartbeat messages. This continues until a termination (or deactivation) is initiated by either the CE or FE. Figure below helps illustrate this state.

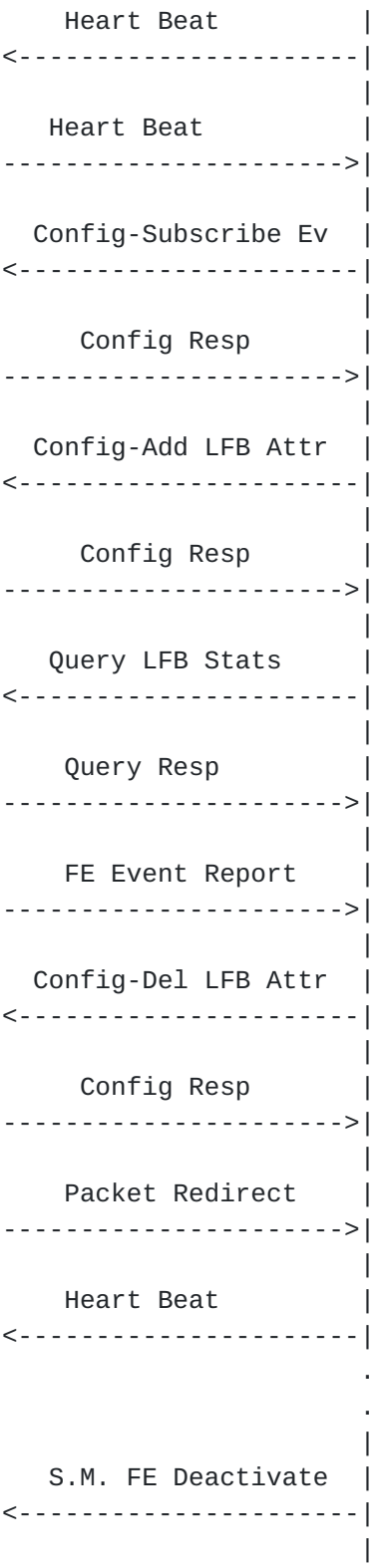


Figure 29: Message exchange between CE and FE during steady-state communication

Note that the sequence of messages shown in the figure serve only as examples and the messages exchange sequences could be different from what is shown in the figure. Also, note that the protocol scenarios described in this section do not include all the different message exchanges which would take place during failover. That is described in the HA section 8.

8. High Availability Support

Editorial Note: This section currently focuses only on CE-CE redundancy. We need to further discuss the FE-FE view. We also need to discuss Multiple Primary CEs.

The ForCES protocol provides mechanisms for CE redundancy and failover, in order to support High Availability. There can be multiple redundant CEs and FEs in a ForCES NE. However, at any time there can only be one Primary CE controlling the FEs and there can be multiple secondary CEs. The FE and the CE PL are aware of the primary and secondary CEs. This information (primary, secondary CEs) is configured in the FE, CE PLs during pre-association by FEM, CEM respectively. Only the primary CE sends Control messages to the FEs. The FE may send its event reports, redirection packets to only the Primary CE (Report Primary Mode) or it may send these to both primary and secondary CEs (Report All Mode). (The latter helps with keeping state between CEs synchronized, although it does not guarantee synchronization.) This behavior or HA Modes are configured during Association setup phase but can be changed by the CE anytime during protocol operation. A CE-to-CE synchronization protocol will be needed in most cases to support fast failover, however this will not be defined by the ForCES protocol.

During a communication failure between the FE and CE (which is caused due to CE or link reasons, i.e. not FE related), the TML on the FE will trigger the FE PL regarding this failure. The FE PL will send a message (Event Report) to the Secondary CEs to indicate this failure or the CE PL will detect this and one of the Secondary CEs takes over as the primary CE for the FE. An explicit message (State Maintenance Move command) from the primary CE, can also be used to change the Primary CE for an FE during normal protocol operation. In order to support fast failover, the FE will establish association (setup msg) as well as complete the capability exchange with the Primary as well as all the Secondary CEs (in all scenarios/modes).

These two scenarios (Report All, Report Primary) have been illustrated in the figures below.

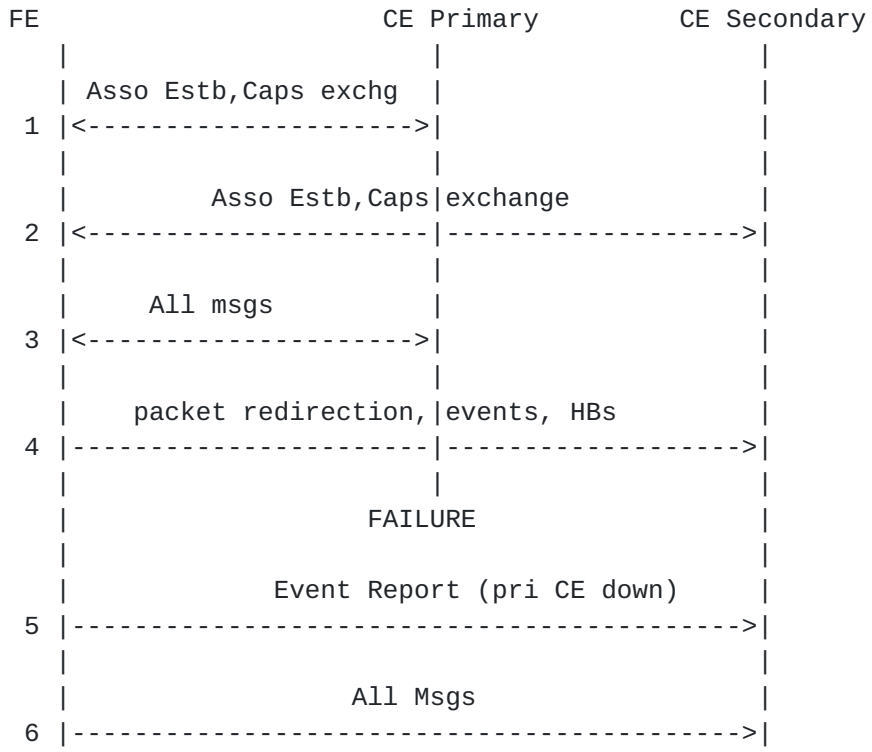


Figure 30: CE Failover for Report All mode

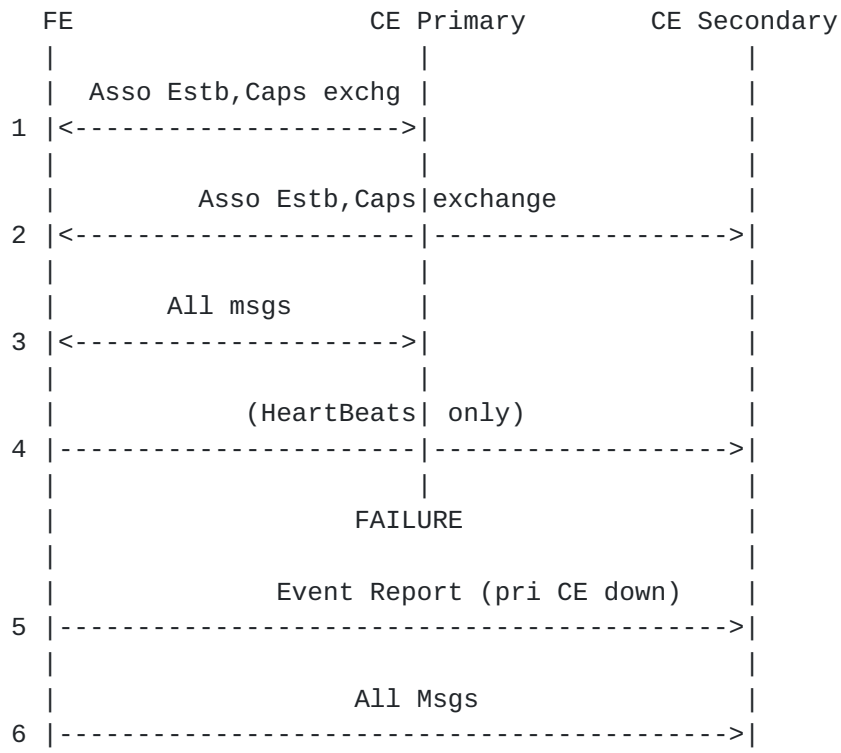


Figure 31: CE Failover for Report Primary Mode

8.1 Responsibilities for HA

TML level - Transport level:
1. The TML controls logical connection availability and failover.
2. The TML also controls peer HA managements.

At this level, control of all lower layers example transport level (such as IP addresses, MAC addresses etc) and associated links going down are the role of the TML.

PL Level:
All the other functionality including configuring the HA behavior during setup, the CEIDs are used to identify primary, secondary CEs, protocol Messages used to report CE failure (Event Report), Heartbeat messages used to detect association failure, messages to change primary CE (state maintenance move), and other HA related operations described before are the PL responsibility.

To put the two together, if a path to a primary CE is down, the TML would take care of failing over to a backup path, if one is available. If the CE is totally unreachable then the PL would be informed and it will take the appropriate actions described before.

9. Security Considerations

ForCES architecture identified several [Reference Arch] levels of security. ForCES PL uses security services provided by the ForCES TML layer. TML layer provides security services such as endpoint authentication service, message authentication service and confidentiality service. Endpoint authentication service is invoked at the time of pre-association connection establishment phase and message authentication is performed whenever FE or CE receives a packet from its peer.

Following are the general security mechanism that needs to be in place for ForCES PL layer.

- Security mechanism are session controlled that is once the security is turned ON depending upon the chosen security level (No Security, Authentication only, Confidentiality), it will be in effect for the entire duration of the session.
- Operator should configure the same security policies for both primary and backup FE's and CE's (if available). This will ensure uniform operations, and to avoid unnecessary complexity in policy configuration.
- ForCES PL endpoints SHOULD pre-established connections with both primary and backup CE's. This will reduce the security messages and enable rapid switchover operations for HA.

9.1 No Security

When No security is chosen for ForCES protocol communication, both endpoint authentication and message authentication service needs be performed by ForCES PL layer. Both these mechanism are weak and does not involve cryptographic operation. Operator can choose "No security" level when the ForCES protocol endpoints are within an single box.

In order to have interoperable and uniform implementation across various security levels, each CE and FE endpoint MUST implement this level. The operations that are being performed for "No security" level is required even if lower TML security services are being used.

9.1.1 Endpoint Authentication

Each CE and FE PL layer maintain set of associations list as part of configuration. This is done via CEM and FEM interfaces. FE MUST connect to only those CE's that are configured via FEM similarly CE should accept the connection and establish associations for the FE's which are configured via CEM. CE should validate the FE identifier before accepting the connection during the pre-association phase.

9.1.2 Message authentication

When CE or FE generates initiates a message, the receiving endpoint MUST validate the initiator of the message by checking the common header CE or FE identifiers. This will ensure proper protocol functioning. We recommend this extra step processing even if the underlying TLM layer security services.

9.2 ForCES PL and TML security service

This section is applicable if operator wishes to use the TML security services. ForCES TML layer MUST support one or more security service such as endpoint authentication service, message authentication service, confidentiality service as part of TML security layer functions. It is the responsibility of the operator to select appropriate security service and configure security policies accordingly. The details of such configuration is outside the scope of ForCES PL and is depending upon the type of transport protocol, nature of connection.

All these configurations should be done prior to starting the CE and FE.

When certificates-based authentication is being used at TML layer, the certificate can use ForCES specific naming structure as certificate names and accordingly the security policies can be configured at CE and FE.

9.2.1 Endpoint authentication service

When TML security services are enabled. ForCES TML layer performs endpoint authentication. Security association is established between CE and FE and is transparent to the ForCES PL layer.

We recommend that FE after establishing the connection with the primary CE, should establish the security association with the backup CE (if available). During the switchover operation CE's security state associated with each SA's are not transferred. SA between primary CE and FE and backup CE and FE are treated as two separate SA's.

9.2.2 Message authentication service

This is TML specific operation and is transparent to ForCES PL layer[TML document].

9.2.3 Confidentiality service

This is TML specific operation and is transparent to ForCES PL layer.[TML document]

10. References

10.1 Normative References

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC3654] Khosravi, H. and T. Anderson, "Requirements for Separation of IP Control and Forwarding", RFC 3654, November 2003.

[RFC3746] Yang, L., Dantu, R., Anderson, T. and R. Gopal, "Forwarding and Control Element Separation (ForCES) Framework", RFC 3746, April 2004.

10.2 Informational References

[FE-MODEL] Yang, L., "ForCES Forwarding Element Model", Feb. 2004.

Author's Address

Avri Doria
ForCES Protocol Design Team

Phone: +1 401 663 5024
EMail: avri@acm.org

Appendix A. Individual Authors/Editors Contact

The participants in the ForCES Protocol Team:

Author	Email
Ligang Dong	donglg@mail.hzic.edu.cn
Avri Doria	avri@acm.org
Ram Gopal	ram.gopal@nokia.com
Robert Haas	rha@zurich.ibm.com
Jamal Hadi Salim	hadi@znyx.com
Hormuzd M Khosravi	hormuzd.m.khosravi@intel.com
Weiming Wang	wmwang@mail.hzic.edu.cn

Table 1

Appendix B. IANA considerations

tbd

Appendix C. Implementation Notes

C.1 TML considerations

Having separated the PL from the TML layer, it became clear that the TML layer needed to understand the desires of the PL layer to service it. Example: How does the TML layer map prioritization or reliability needs of a PL message? To see the challenge involved, assume that all of the FE TML, FE PL, CE TML and CE PL are implemented by different authors probably belonging to different organizations. Three implementation alternatives were discussed.

As an example, consider a TML which defines that PL messages needing reliability get sent over a TCP connection; then TML-PL interfaces are:

- PL to call a special API: example send_reliable(msg) which is translated by the TML to mean send via TCP.
- PL to call a generic API: example send(msg) with explicit msg flags turned to say "reliability needed" and the TML translates this to mean send via TCP.
- PL sends the Forces Messages such a message is inferred to mean send via TCP by the TML.

in #1 and #2 the msg includes a ForCES msg with metadata flags which are consumed by the TML layer.

#3 is a technique that will be referred as inference-by-TML technique. It simplifies the standardization effort since both #1 and #2 will require standardization of an API. Two ideas discussed for TML inference of PL messages are:

1. Looking at the flags in the header.
2. Looking at the message type.

#1 and #2 can still be used if a single organization implements both (PL and TML) layers. It is also reasonable that one organization implements the TML and provides an abstraction to another organization to implement a PL layer on.

C.1.1 PL Flag inference by TML

1. Reliability
   This could be "signalled" from the PL to the TML via the ACK flag. The message type as well could be used to indicate this.
2. No reliability
   Could be signalled via missing ACK flag. The message type as well could be used to indicate this.
3. Priorities
   A remapping to be defined via the FEM or the CEM interface depending on the number of TML priorities available.

4. Addressing
   This is TML specific. For example a TML that is capable of multicast transport may map a multicast PL ID to a multicast transport address.
5. Event notifications
   The TML must be able to send to the PL notifications.
   1. The TML should be able to send Transport level congestion notifications to the PL.
   2. Link events for HA purposes if configuration requires it
   3. Events that will trigger PL layer events from the TML. As an example, an HA event at the TML layer like a failure of CE detected at TML on the FE may belong to this. In this case, a PL event msg will be triggered and sent to CE.
   4. Events that are intrinsic to the same CE or FE a TML is located. These will not trigger any PL msg, instead, they just act as notification to PL core (FE object). The congestion event generated at the transmission source side may belong to this, because it usually only needs to tell the upper PL at the same side rather than the opposite side that congestion has happened along the path. E.g., a congestion event at CE TML layer only need to tell CE PL of this, rather than the opposite FE via a PL msg.

C.1.2 Message type inference to Mapping at the TML

In this case one would define the desires of the different message types and what they expect from the TML. For example:

1. Association Setup, Teardown, Config, Query the PL will expect the following services from TML: Reliable delivery and highest prioritization.
2. Packet Redirect, HB Message Types, and Event Reports the PL will require the following services from TML: Medium Prioritization, and notifications when excessive losses are reached.

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.