XCON Working Group
Internet-Draft
Expires: January 4, 2005

G. Camarillo
Ericsson
J. Ott
Universitaet Bremen
K. Drage
Lucent Technologies
July 6, 2004

The Binary Floor Control Protocol (BFCP)
draft-ietf-xcon-bfcp-00.txt

Status of this Memo

Copyright Notice

Abstract

This document specicies the Binary Floor Control Protocol (BFCP). BFCP is used between conference participants and floor control servers, and between floor chairs (i.e., moderators) and floor control servers.

Table of Contents

1. Introduction

Some applications (e.g., conference applications) need to manage the access to a set of shared resources, such as the right to send media over a particular media stream. Floor control enables such applications to provide users with safe access to these resources.

The Requirements for Floor Control Protocol [15] list a set of requirements that need to be met by floor control protocols. The Binary Floor Control Protocol (BFCP), which is specified in this document, meets these requirements.

Section 2 defines the terminology used throughout this document and Section 3 discusses the scope of BFCP (i.e., which tasks fall within the scope of BFCP and which ones are performed using different mechanisms). Section 4 provides a non-normative overview of BFCP operation and subsequent sections provide the normative specification of BFCP.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [2] and indicate requirement levels for compliant implementations.

Conference Policy: The complete set of rules for a particular conference manipulated by the conference policy server. It includes the membership policy and the media policy. There is an instance of conference policy for each conference.

Conference Policy Control Protocol (CPCP): The protocol used by clients to manipulate the conference policy.

Floor: A permission to temporarily access or manipulate a specific shared resource or set of resources.

Floor chair: A user (or an entity) who manages one floor (grants, denies, or revokes a floor). The floor chair does not have to be a member in a conference.

Floor control: A mechanism that enables applications or users to gain safe and mutually exclusive or non-exclusive input access to the shared object or resource.

Floor control server: A logical entity that maintains the state of the floor(s) including which floors exists, who the floor chairs are,

who holds a floor, etc. Requests to manipulate a floor are directed at the floor control server.

Media Policy: A set of rules manipulated by the conference policy server regarding the media composition of the conference. The media policy is used by the focus to determine the mixing characteristics for the conference. The media policy includes rules about which participants receive media from which other participants, and the ways in which that media is combined for each participant. In the case of audio, these rules can include the relative volumes at which each participant is mixed. In the case of video, these rules can indicate whether the video is tiled, whether the video indicates the loudest speaker, and so on.

3. Scope

Figure 1 shows the tasks that BFCP can perform.

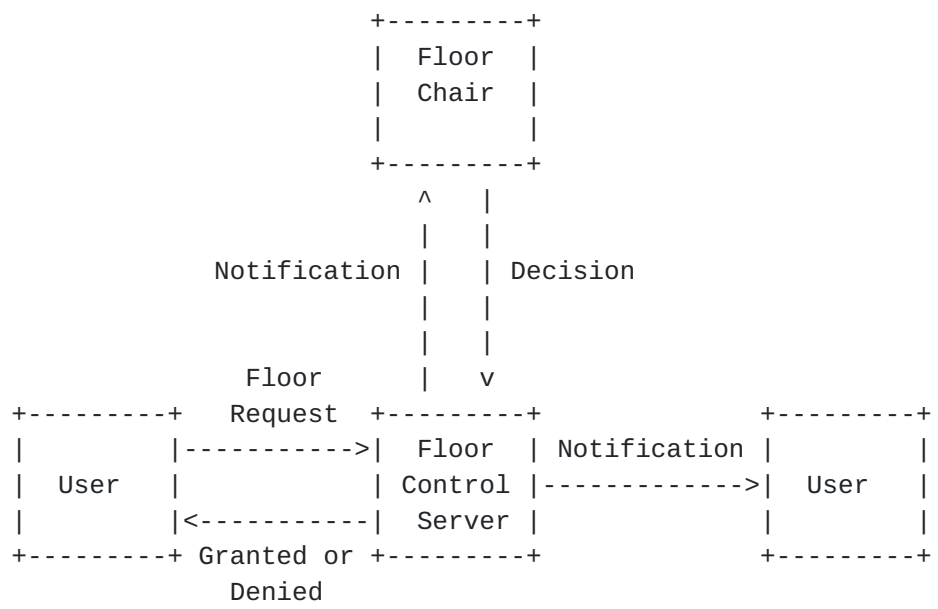


Figure 1: Functionality provided by BFCP

BFCP provides a means:

- for users to send floor requests to floor control servers.
- for floor control servers to grant or deny requests to access a given resource from users.
- for floor chairs to send floor control servers decisions regarding floor requests.
- for floor control servers to keep users and floor chairs informed about the status of a given floor.

Even though tasks that do not belong to the previous list are outside the scope of BFCP, some of these out-of-scope tasks relate to floor control and are essential to create floors and to establish BFCP connections between different entities. In the following sections, we discuss some of these tasks and mechanisms to perform them.

3.1 Floor Creation

The conference policy for a particular conference contains the floors of the conference and the resource or resources associated to each floor. For example, a conference may have two floors: one controlling who can talk at a particular time and another controlling who can write on a shared whiteboard.

According to the definitions in Section 2, the conference policy is manipulated using a Conference Policy Control Protocol (CPCP), such as [16]. Consequently, floor creation and termination is handled by CPCP. In addition, CPCP also handles the association of a given floor with a resource or a set of resources (e.g., media streams).

Additionally, the floor control server needs to stay up to date on changes on the conference policy (e.g., when a new floor is created). The floor control may use a mechanism such as the XCAP event package [19] to keep itself up to date.

3.2 Obtaining Information to Contact a BFCP Floor Server

A user or a floor chair needs a set of data in order to establish a BFCP connection to a floor control server. These data include the transport address of the server, the conference identifier, and the user identifier.

Users can obtain this information in different ways. Two possibilities are using CPCP and using the offer/answer [11] exchange which is used to establish media streams between the user and the conference server. Section 15 discusses how to use an SDP [5] offer/answer [11] exchange to obtain this information.

3.3 Generating a Shared Secret

Authentication and integrity protection in BFCP are based on a shared secret between the user or floor chair, and the floor control server. So, there is a need for a mechanism to generate such a shared secret.

When the user or the floor control chair obtains a the information needed to contact the BFCP floor server over a secure channel (e.g., an offer/answer exchange using SIP [10] protected using S/MIME), they can get the shared secret using the same channel.

If there is no secure channel available, a different mechanism needs to be used. For example, MIKEY [18] allows an offerer and an answerer to perform a Diffie-Hellman key exchange.

3.4 Obtaining Floor-Resource Associations

Floors are associated to resources. For example, a floor that controls who talks at a given time has a particular audio stream as its associated resource. Associations between floors and resources are part of the conference policy, which is manipulated using CPCP.

Users and floor chairs need to know which resources are associated to which floors. They can obtain this information using different mechanisms, such as CPCP or an offer/answer [11] exchange. Section 15 describes how to use an offer/answer exchange to obtain these associations.

> Note that users perform offer/answer exchanges with the SIP focus of the conference. So, the SIP focus needs to obtain information about associations between floors and resources using a mechanism such as CPCP in order to be able to provide this information to a user in an offer/answer exchange.

3.5 Policy Enforcement

A user whose floor request is granted has the right to use in a certain way the resource or resources associated to the floor that was requested. For example, the user may have the right to talk (i.e., send media over a particular audio stream).

Nevertheless, holding a floor does not imply that others will not be able to use its associated resources at the same time, even if they do not have the right to do so. According to the definition in Section 2, the media policy of a conference is the one that determines which users can actually use the resources in the conference.

So, if the policy of a conference is to enforce floor control decisions, every change in the status of any floor needs to be reflected in the media policy of the conference. For example, the mixer only accepts media from the user who holds the floor.

A way to reflect the status of the floors in the media policy is to have the floor control server manipulate the media policy using CPCP. Nevertheless, there are another ways to enforce floor control policies, such as having a floor control chair manipulate the media policy (using CPCP) only if there are misbehaving users trying to use a resource without holding its associated floor.

4. Overview of Operation

This section provides a non-normative description of BFCP operations. Section 4.1 describes the interface between users and floor control servers and Section 4.2 describes the interface between floor chairs and floor control servers

BFCP messages, which use a TLV (Type-Length-Value) binary encoding, which consist of a common header followed by a set of TLVs. The common header contains, among other information, a 32-bit conference identifier. Users and floor chairs are identified by a 16-bit user identifier, which is carried in a TLV.

4.1 User - Floor Control Server Interface

Users request a floor by sending a FloorRequest message to the floor control server. BFCP supports third party floor requests. That is, the user sending the floor request need not be the same as the user who will get the floor once the floor request is granted. FloorRequest messages carry the identity of the requester in a USER-ID TLV, and the identity of the beneficiary of the floor, in third party floor requests, in a BENEFICIARY-ID TLV.

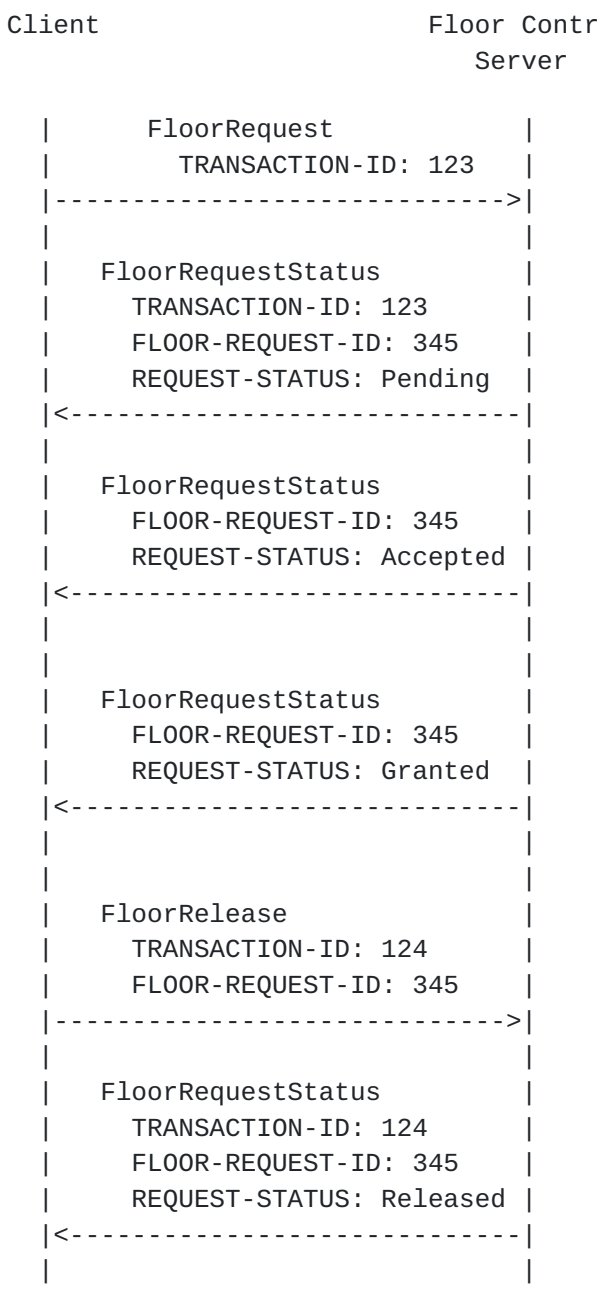
> Third party floor requests can be sent by users that have a BFCP connection to the floor control server, but who are not conference participants (i.e., they do not handle any media).
> XXX: Users need a means to authorize 3rd parties to request floors on their behalf. CPCP?

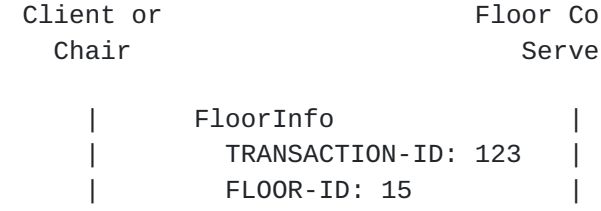
FloorRequest messages identify the floor or floors being requested by carrying their 16-bit floor identifiers in FLOOR-ID TLVs. If a FloorRequest message carries more than one floor identifier, the floor control server treats all the floor requests as an atomic package. That is, the floor control server either grants or denies all the floors in the FloorRequest message.

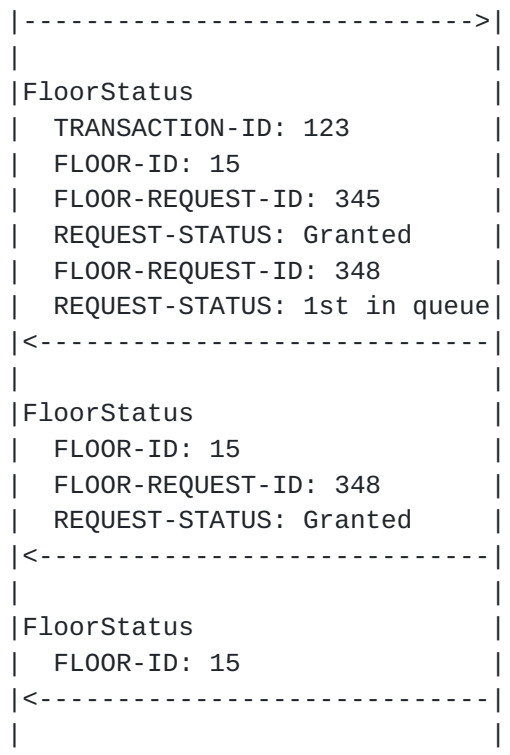
Users can request privacy by including a PRIVACY TLV in their FloorRequest messages. In this case, the floor control server does not disclose the identity of the requester of the floor to the rest of the users and floor chairs.

Floor control servers respond to FloorRequest messages with FloorStatus messages.

Editor's note: This section will be finished when we have agreed on what it is specified in the rest of this document. For the time being, below there are two typical call flows: a client requesting a floor and a client requesting information about a floor.

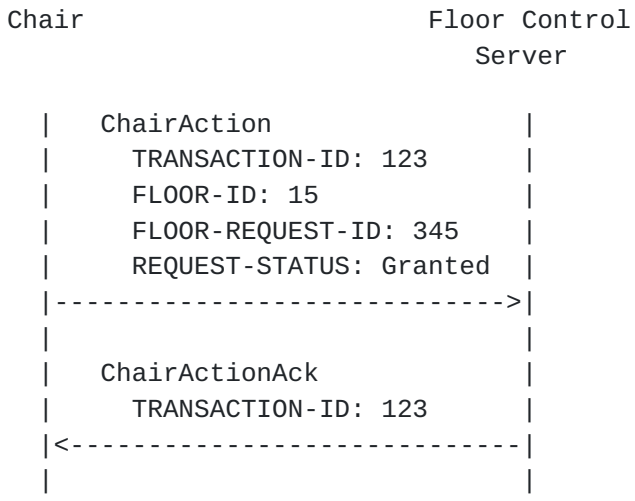

Client
Floor Control Server
FloorRequest
TRANSACTION-ID: 123
FloorRequestStatus
TRANSACTION-ID: 123
FLOOR-REQUEST-ID: 345
REQUEST-STATUS: Pending
FloorRequestStatus
FLOOR-REQUEST-ID: 345
REQUEST-STATUS: Accepted
FloorRequestStatus
FLOOR-REQUEST-ID: 345
REQUEST-STATUS: Granted
FloorRelease
TRANSACTION-ID: 124
FLOOR-REQUEST-ID: 345
FloorRequestStatus
TRANSACTION-ID: 124
FLOOR-REQUEST-ID: 345
REQUEST-STATUS: Released


Client or Chair
Floor Control Server
FloorInfo
TRANSACTION-ID: 123
FLOOR-ID: 15



4.2 Floor Chair - Floor Control Server Interface

TBD. For the time being, below there is the typical call flow for this interface.

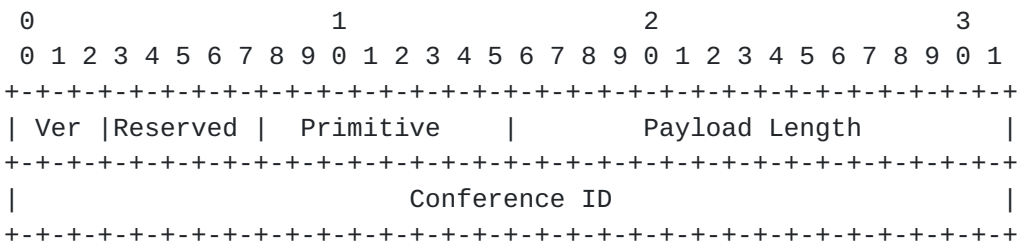


5. Packet Format

BFCP packets consist of an 8-byte fixed header followed by attributes. All the protocol values MUST be sent in network byte order.

5.1 FIXED-HEADER Format

The following is the FIXED-HEADER format.



Ver: the 3-bit version field MUST be set to 1 to indicate this version of BFCP.

Reserved: at this point, the 5 bits in the reserved field SHOULD be set to zero by the sender of the message and MUST be ignored by the receiver.

Primitive: this 8-bit field identifies the main purpose of the message. The following primitive values are defined:

Value	Primitive	Direction
0	FloorRequest	C -> S
1	FloorRelease	C -> S
2	FloorRequestInfo	S -> C ; Ch -> S
3	FloorRequestStatus	S -> C
4	FloorInfo	C -> S ; Ch -> S
5	FloorStatus	S -> C ; S -> Ch
6	ChairAction	Ch -> S
7	ChairActionAck	S -> Ch
8	Ping	C -> S ; Ch <-> S
9	PingAck	S -> C ; Ch <-> S
10	Error	S -> C ; S -> Ch

S: Server C: Client Ch: Chair

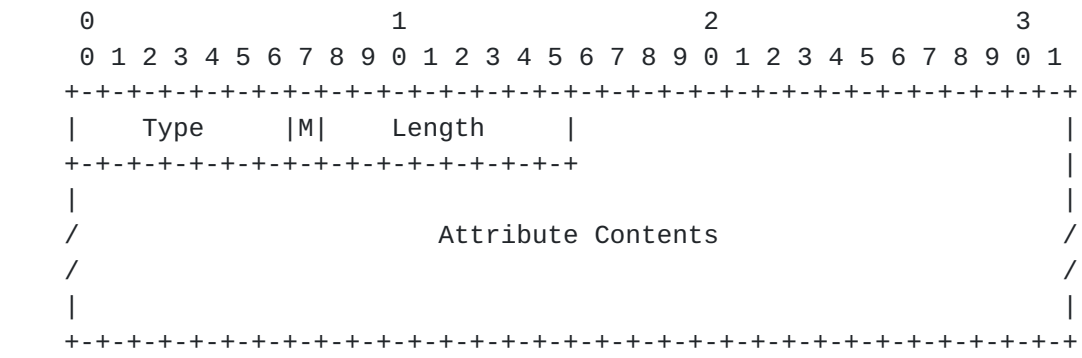
Payload Length: this 16-bit field contains length of the message in

4-byte units excluding the fixed header.

Conference ID: this 32-bit identifies the conference the message belongs to.

5.2 Attribute Format

BFCP attributes are encoded in TLV (Type-Length-Value) format. TLVs are 32-bit aligned.



Type: this 7-bit field contains the code for the attribute.

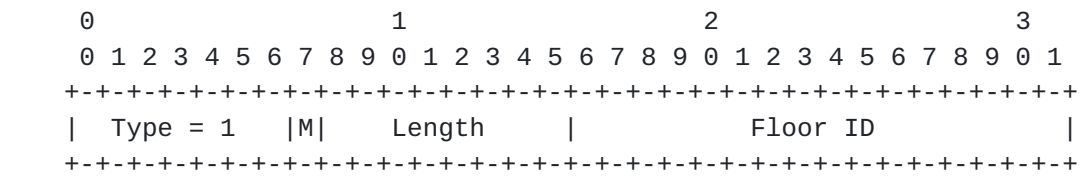
M: the 'M' bit, known as the Mandatory bit, indicates whether support of the attribute is required. If an unrecognized attribute with the 'M' bit set is received, the message is rejected.

Length: this 8-bit field contains the length of the attribute in bytes, excluding any padding defined for specific attributes. The Type, 'M' bit, and Length fields are included.

Attribute Contents: the contents of the different TLVs are defined in the following sections.

5.2.1 FLOOR-ID

The following is the format of the contents of the FLOOR-ID attribute, whose attribute type is 1.

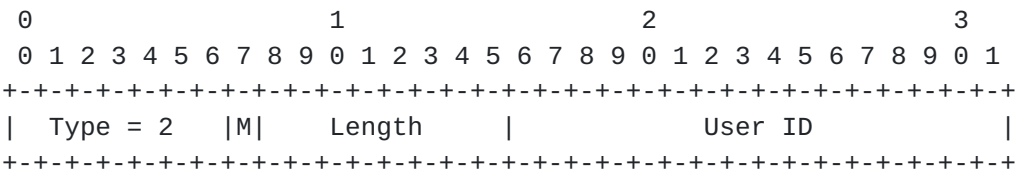


Floor ID: this field contains a 16-bit value that uniquely identifies

a floor within a conference.

5.2.2 USER-ID

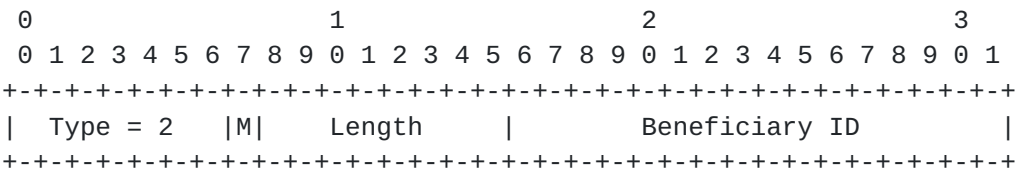
The following is the format of the contents of the USER-ID attribute, whose attribute type is 2.



User ID: this field contains a 16-bit value that uniquely identifies a user within a conference.

5.2.3 BENEFICIARY-ID

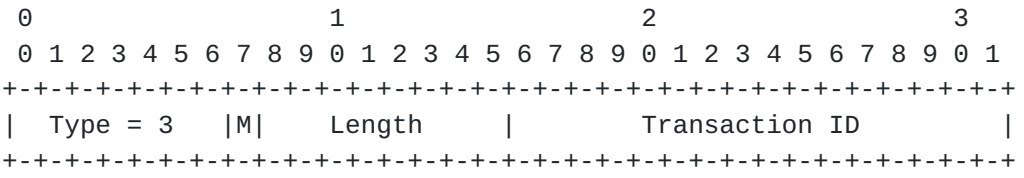
The following is the format of the contents of the BENEFICIARY-ID attribute, whose attribute type is 2.



Beneficiary ID: this field contains a 16-bit value that uniquely identifies a user within a conference.

5.2.4 REQUEST-ID

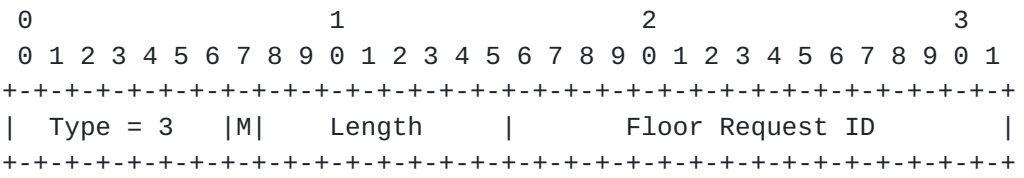
The following is the format of the contents of the TRANSACTION-ID attribute, whose attribute type is 3.



Transaction ID: this field contains a 16-bit value that allows users to match a given message with its response.

5.2.5 FLOOR-REQUEST-ID

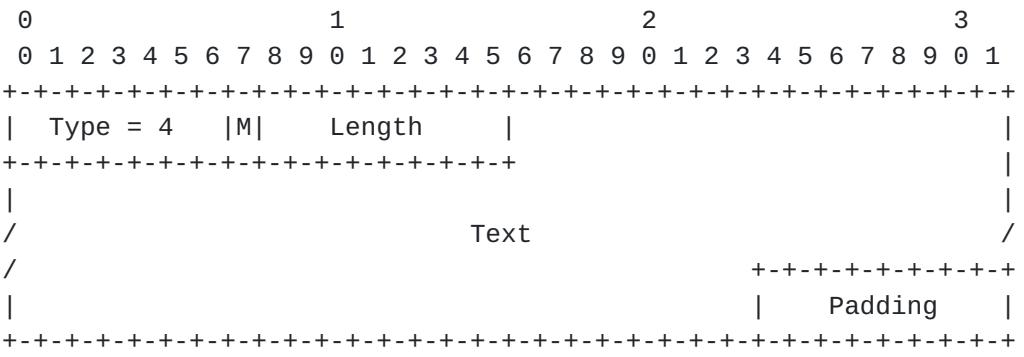
The following is the format of the contents of the FLOOR-REQUEST-ID attribute, whose attribute type is 4.



Floor Request ID: this field contains a 16-bit value that indentifies a floor request at the floor control server.

5.2.6 HUMAN-READABLE-INFO

The following is the format of the contents of the HUMAN-READABLE-INFO attribute, whose attribute type is 5.



Text: this field contains UTF-8 [13] encoded text.

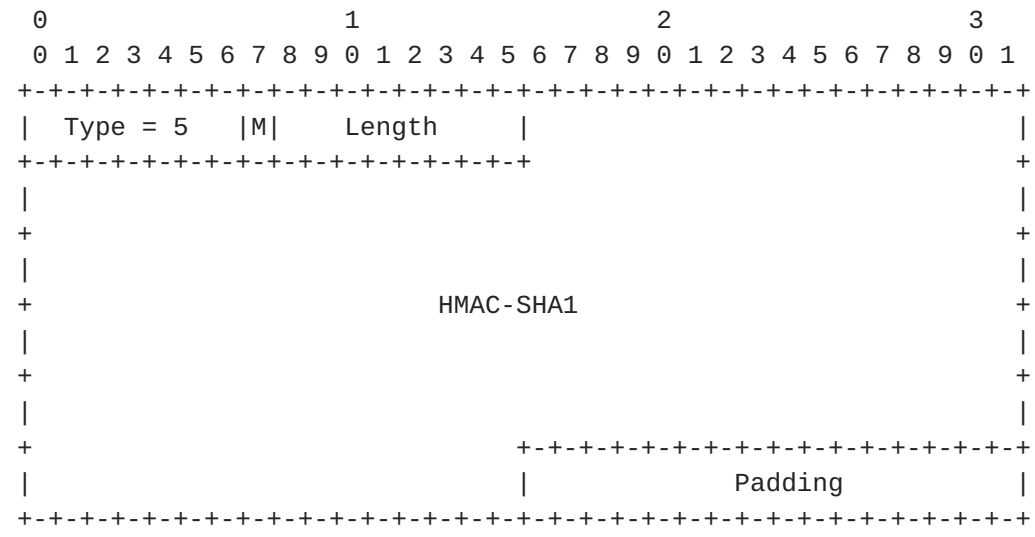
In some situations, the contents of the Text field may be generated by an automata. If such automata has information about the preferred language of the receiver of a particular HUMAN-READABLE-INFO TLV, it may use this language to generate the Text field.

Padding: one, two, or three bytes of padding added so that the contents of the HUMAN-READABLE-INFO TLV is 32-bit aligned. The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver. If the TLV is already 32-bit aligned, no padding is needed.

5.2.7 INTEGRITY

The following is the format of the contents of the INTEGRITY

attribute, whose attribute type is 6.

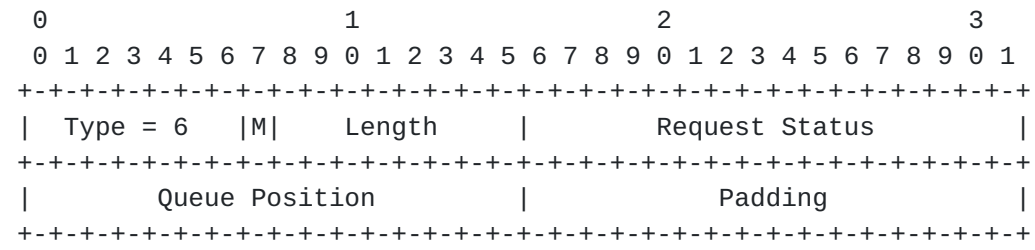


HMAC-SHA1: this 20-byte field contains an HMAC-SHA1 [1] of the BFCP message. Its calculation is described in Section 11.

Padding: two bytes of padding added so that the contents of the HMAC-SHA1 TLV is 32-bit aligned. The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver.

5.2.8 REQUEST-STATUS

The following is the format of the contents of the REQUEST-STATUS attribute, whose attribute type is 7.



Request Status: this 16-bit field contains the status of the request, as described in the following table.

Value	Status
0	Pending

1	Accepted
2	Granted
3	Denied
4	Cancelled
5	Released
6	Revoked
7	Replaced

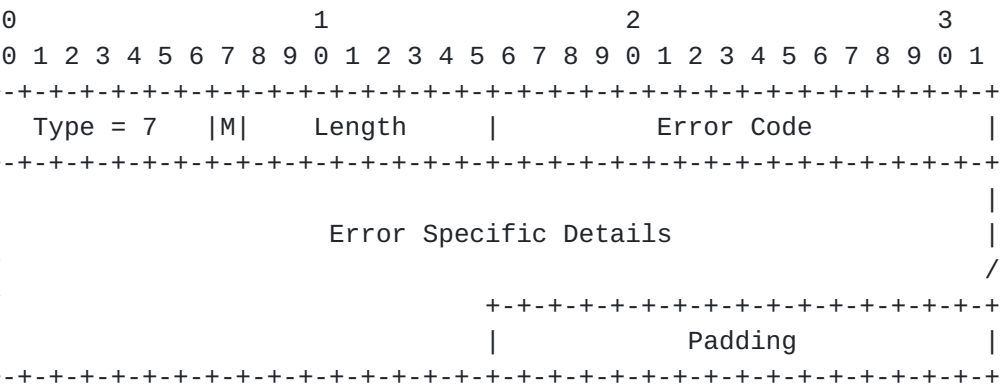
Queue Position: this 16-bit field contains, when applicable, the position of the floor request in the floor request queue at the server. If the Request Status value is different from Accepted, the floor control server does not implement a floor request queue, or the floor control server does not want to provide the client with this information, all the bits of this field SHOULD be set to zero.

A floor request is in Pending state if the floor control server needs to contact a floor chair in order to accept the floor request, but has not done it yet. Once the floor control chair accepts the floor request, the floor request is moved to the Accepted state.

Padding: two bytes of padding added so that the contents of the REQUEST-STATUS TLV is 32-bit aligned. The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver.

5.2.9 ERROR-CODE

The following is the format of the contents of the ERROR-CODE attribute, whose attribute type is 8.

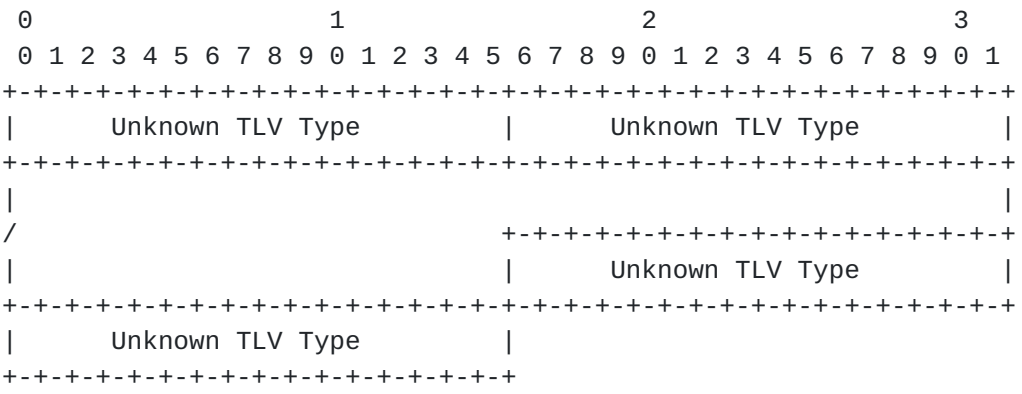


Error Code: this 16-bit field contains an error code from the following table.

Value	Meaning

0	Conference does not Exist
1	Authentication Failed
2	Unknown Mandatory TLV
3	Floor Request ID Does Not Exist
4	Unauthorized Operation
5	User does not Exist
6	Invalid Request-ID
7	INTEGRITY TLV Required

Error Specific Details: Present only for certain Error Codes. In this document, only for Error Code 2 (Unknown Mandatory TLV). For Error Code 2, this field contains the Types of the TLVs (which were present in the message that triggered the Error message) that were unknown to the receiver, encoded as follows.



Padding: one, two, or three bytes of padding added so that the contents of the ERROR-CODE TLV is 32-bit aligned. If the TLV is already 32-bit aligned, no padding is needed.

The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver. Note all the Error Codes defined in this document but Error Code 2, result in a TLV which is already 32-bit aligned (i.e., no need of padding). Error Code 2 results in a TLV that may need 2 bytes of padding.

5.2.10 USER-DISPLAY-NAME

The USER-DISPLAY-NAME attribute type is 9 and its format is the same as the HUMAN-READABLE-INFO attribute. The Text field in the USER-DISPLAY-NAME attribute contains the name of the user.

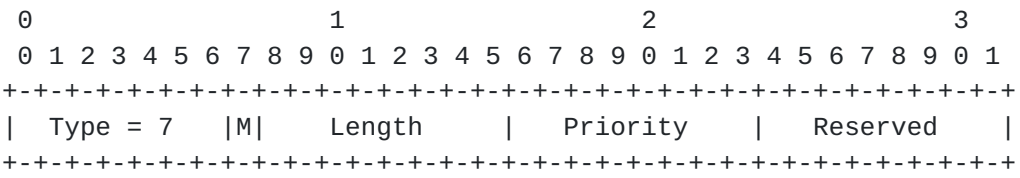
5.2.11 USER-URI

The USER-URI attribute type is 10 and its format is the same as the HUMAN-READABLE-INFO attribute. The Text field in the USER-URI

attribute contains the URI of the user.

5.2.12 PRIORITY

The following is the format of the contents of the PRIORITY attribute, whose attribute type is 11.

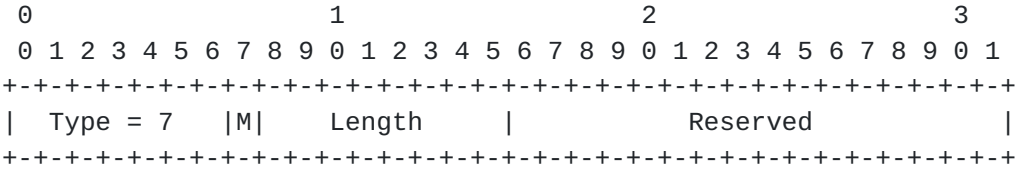


Priority: the higher the 8-bit value, the more priority is requested for a given floor request.

Reserved: at this point, the 8 bits in the reserved field SHOULD be set to zero by the sender of the message and MUST be ignored by the receiver.

5.2.13 PRIVACY

The following is the format of the contents of the PRIVACY attribute, whose attribute type is 12.



Reserved: at this point, the 16 bits in the reserved field SHOULD be set to zero by the sender of the message and MUST be ignored by the receiver.

6. Message Format

This section contains the ABNF [3] of the BFCP messages.

6.1 FloorRequest

Clients request a floor by sending a FloorRequest message to the floor control server. In addition, the FloorRequest message is also used to modify existing floor requests. The following is the format of the FloorRequest message:

```
FloorRequest =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
                 [BENEFICIARY-ID]
                 [PRIVACY]
                 [FLOOR-REQUEST-ID]
                *(FLOOR-ID)
                 [HUMAN-READABLE-INFO]
                 [PRIORITY]
                 [INTEGRITY]
```

6.2 FloorRelease

Clients release a floor by sending a FloorRelease message to the floor control server. Clients also use the FloorRelease message to cancel pending floor requests. The following is the format of the FloorRelease message:

```
FloorRelease =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
                 (FLOOR-REQUEST-ID)
                 [INTEGRITY]
```

6.3 FloorRequestInfo

Clients request information about a floor request by sending a FloorRequestInfo message to the floor control server. The following is the format of the FloorRequest message:

```
FloorRequestInfo =   (FIXED-HEADER)
                     (TRANSACTION-ID)
                     (USER-ID)
                     [BENEFICIARY-ID]
                     [FLOOR-REQUEST-ID]
                     [INTEGRITY]
```

6.4 FloorRequestStatus

The floor control server informs clients about the status of their floor requests by sending them FloorRequestStatus messages. The following is the format of the FloorRequestStatus message:

```
FloorRequestStatus =   (FIXED-HEADER)
                       (TRANSACTION-ID)
                       (USER-ID)
                       [BENEFICIARY-ID]
                       [USER-DISPLAY-NAME]
                       [USER-URI]
                   1*(  (FLOOR-REQUEST-ID)
                      1*(FLOOR-ID)
                        [HUMAN-READABLE-INFO]
                        [PRIORITY]
                        (REQUEST-STATUS)     )
                        [INTEGRITY]
```

6.5 FloorInfo

Clients request information about a floor or floors by sending a FloorInfo message to the floor control server. The following is the format of the FloorRequest message:

```
FloorInfo =   (FIXED-HEADER)
              (TRANSACTION-ID)
              (USER-ID)
             *(FLOOR-ID)
              [INTEGRITY]
```

6.6 FloorStatus

The floor control server informs clients about the holder of a floor by sending them FloorStatus messages. The following is the format of the FloorStatus message:

```
FloorStatus        =     (FIXED-HEADER)
                         [TRANSACTION-ID]
                         (USER-ID)
                         [FLOOR-ID]
                       *( (FLOOR-REQUEST-ID)
                         [BENEFICIARY-ID]
                         [USER-DISPLAY-NAME]
                         [USER-URI]
                        *(FLOOR-ID)
                         [HUMAN-READABLE-INFO]
                         [PRIORITY]
                         (REQUEST-STATUS)      )
                         [INTEGRITY]
```

6.7 ChairAction

Chairs send instructions to floor control servers by sending ChairAction messages. The following is the format of the ChairAction message:

```
ChairAction  =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
               1*(FLOOR-ID)
                 (FLOOR-REQUEST-ID)
                 (REQUEST-STATUS)
                 [HUMAN-READABLE-INFO]
                 [INTEGRITY]
```

6.8 ChairActionAck

Floor control servers condirm that they have accepted a ChairAction message by sending a ChairActionAck message. The following is the format of the ChairActionAck message:

```
ChairActionAck  =   (FIXED-HEADER)
                    (TRANSACTION-ID)
                    (USER-ID)
                    [INTEGRITY]
```

6.9 Ping

Clients check the liveness of servers, and servers of clients, by sending a Ping message. The following is the format of the Ping message:

```
Ping         =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
                 [INTEGRITY]
```

6.10 PingAck

Servers confirm that they are alive on reception of a Ping message by sending a PingAck message. The following is the format of the PingAck message:

```
PingAck      =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
                 [INTEGRITY]
```

6.11 Error

The floor control server informs clients about errors processing requests by sending them Error messages. The following is the format of the Error message:

```
Error             =   (FIXED-HEADER)
                      (TRANSACTION-ID)
                      (USER-ID)
                      (ERROR-CODE)
                      [HUMAN-READABLE-INFO]
                      [INTEGRITY]
```

7. Transport

BFCP entities exchange BFCP messages using TCP connections. TCP provides an in-order reliable delivery of a stream of bytes. Consequently, message framing is implemented in the application layer. BFCP implements application-layer framing using TLVs.

8. Lower-Layer Security

BFCP relies on lower-layer security mechanisms to provide replay protection, and confidentiality. BFCP servers MUST support TLS [4], and clients SHOULD support TLS. Clients and servers MAY support other security mechanisms.

OPEN ISSUE: do we want to implement replay-protection in the protocol instead of relying on TLS?

Servers and clients that implement TLS MUST support XXX (cypher and hash algorithm).

9. Privacy

Clients may not want to disclose their identity to other users when they request a floor. Clients can request privacy by adding a PRIVACY TLV to their FloorRequest messages. This way, a client can request privacy for a particular floor request, and not request it for another one.

Floor control servers do not disclose the identity of the user requesting a floor when privacy is requested (e.g., their FloorRequestStatus messages do not contain a BENEFICIARY-ID TLV). Nevertheless, if an attacker can get access to a given FloorRequest message, it will be able to discover the ID of the user requesting the floor, even if the message contains a PRIVACY TLV.

So, clients that want to avoid this attack need to implement confidentiality underneath BFCP, as described in Section 8.

OPEN ISSUE: shall we define the PRIVACY TLV as a simple flag, or do we want to have more options. For example, the floor control can disclose the user's identity to the chair, but not to the rest of the users.

10. Protocol Transactions

In BFCP, there are two types of transactions: client-initiated transactions and server-initiated transactions. Client-initiated transactions consist of a request from a client to a server and a response from the server to the client. The request carries a TRANSACTION-ID TLV which the server copies into the response. Clients use Transaction ID values to match responses with previously-issued requests.

Server-initiated transactions consist of a single message from a server to a client. Consequently, since they do not trigger any response, server-initiated transactions do not have Transaction IDs associated to them.

10.1 Client Behavior

A client starting a client-initiated transaction MUST set the Conference ID in the FIXED-HEADER of the message to the Conference ID for the conference that the client obtained previously.

The client MUST set the Transaction ID value in the TRANSACTION-ID TLV to a number which MUST NOT be reused in another message from the client until a response from the server is received. The client uses the Transaction ID value to match this FloorRequest message with the response from the floor control server.

10.2 Server Behavior

A server sending a response within a client-initiated transaction MUST copy the Conference ID and the TRANSACTION-ID TLV from the request received from the client into the response. Server-initiated transactions MUST NOT contain a TRANSACTION-ID TLV.

11. Authentication

BFCP uses the INTEGRITY TLV to provide client and server authentication, and message integrity. The INTEGRITY TLV contains an HMAC-SHA1 [1] of the BFCP message. The use of SHA1 implies that the length of the HMAC is 20 bytes. The text used as input to HMAC is the BFCP message, including the FIXED-HEADER, up to and including the TLV preceding the INTEGRITY TLV. This text is then padded with zeroes so as to be a multiple of 64 bytes. As a result, the INTEGRITY TLV MUST be the last attribute in any BFCP message. The key used as input to HMAC is the secret shared between the server and the user identified by the USER-ID TLV in the message.

11.1 Client Behavior

Clients SHOULD add an INTEGRITY TLV to all their messages. Furthermore, if a client generates a message without this TLV and receives an Error response with Error Code 7 (INTEGRITY TLV Required), the client SHOULD NOT send more messages without the INTEGRITY TLV to the same server within the same conference.

When a client receives a BFCP message with an INTEGRITY TLV, it calculates the HMAC-SHA1 [1] of the message excluding the INTEGRITY TLV. The key used as input to HMAC is the secret shared between the server and the user identified by the USER-ID TLV in the message. If the resulting value is the same as the one in the INTEGRITY TLV, authentication is considered successful. If the resulting value is different than the one in the INTEGRITY TLV, authentication is considered to have failed and the message MUST NOT be processed further.

11.2 Server Behavior

Servers SHOULD add an INTEGRITY TLV to all their messages. Furthermore, if a client sends messages with this TLV to a server, the server MUST include it as well in its messages to this client.

If a client does not add an INTEGRITY TLV to a message, the server MAY request its addition by returning an Error message with Error Code 7 (INTEGRITY TLV Required).

When a server receives a BFCP message with an INTEGRITY TLV, it calculates the HMAC-SHA1 [1] of the message excluding the INTEGRITY TLV. The key used as input to HMAC is the secret shared between the server and the user identified by the USER-ID TLV in the message. If the resulting value is the same as the one in the INTEGRITY TLV, authentication is considered successful.

If the resulting value is different than the one in the INTEGRITY TLV, authentication is considered to have failed, in which case the server SHOULD return an Error message with Error Code 1 (Authentication Failed). Messages that cannot be authenticated MUST NOT be processed further.

12. Client Operations

This section specifies how clients can perform different operations, such as requesting a floor, using the protocol elements described in earlier sections. Chair operations, such as instructing the floor server to grant or revoke a floor, are described in Section 13.

12.1 Requesting a Floor

A client that wishes to request one or more floors does so by sending a FloorRequest message to the floor control server. The ABNF in Section 6.1 describes the TLVs that a FloorRequest message can contain. In addition, the ABNF specifies which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the client follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV).

The client must insert a USER-ID TLV, which will be used by the server to authenticate and authorize the request. If the user sending the FloorRequest message (identified by the USER-ID TLV) is not the same as the user requesting the floor (i.e., a third party floor request), the client SHOULD add a BENEFICIARY-ID TLV to the message identifying the beneficiary of the floor.

The client must insert at least one FLOOR-ID TLV in the FloorRequest message. If the client inserts more than one FLOOR-ID TLVs, the server will treat all the floor requests as an atomic package. That is, the floor control server will either grant or deny all the floors in the FloorRequest message.

The client may use a HUMAN-READABLE-INFO TLV to state the reason why the floor or floors are being requested. The Text field in the HUMAN-READABLE-INFO TLV is intended for human consumption.

The client may request the server to handle the floor request with a certain priority using a PRIORITY TLV.

12.1.1 Receiving a Response

A message from the server is considered to be a response to the FloorRequest message if the message from the server has the same Conference ID and Transaction ID as the FloorRequest message, as described in Section 10.1.

If the response is a FloorRequestStatus message, the client obtains information about the status of the FloorRequest in a REQUEST-STATUS TLV. If the Request Status value is Granted, the client has been granted all the floors that were requested in the FloorRequest message. If the Request Status value is Denied, the client has been denied all the floors that were requested in the FloorRequest message. The HUMAN-READABLE-INFO TLV, if present, provides extra information which the client MAY display to the user.

A floor request is considered to be ongoing while it is in the Pending, Accepted, or Granted states. FloorRequestStatus messages carrying any of these states will contain a FLOOR-REQUEST-ID TLV. The value of this TLV is used to match subsequent messages received at the client side (e.g., further FloorRequestStatus messages) or at the server side (e.g., a FloorRelease message) with this floor request.

If the response is an Error message, the server could not process the FloorRequest message for some reason, which is described in the Error message.

12.2 Modifying a Floor Request

Clients modify the HUMAN-READABLE-INFO and the PRIORITY attributes of ongoing floor requests by sending a FloorRequest message. The ABNF in Section 6.1 describes the TLVs that a FloorRequest message can contain. In addition, the ABNF specifies which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the client follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV). The client MUST insert a USER-ID TLV, which will be used by the server to authenticate and authorize the request.

FloorRequest messages sent to modify an existing floor request MUST NOT contain any BENEFICIARY or FLOOR-ID TLV. Instead, the client MUST insert a FLOOR-REQUEST-ID TLV identifying the floor request to be modified.

Clients modifying a floor request can provide new HUMAN-READABLE INFO and PRIORITY TLVs but cannot modify the beneficiary of the floor request or the floors being requested.

The client may use a HUMAN-READABLE-INFO TLV to state the reason why the floor or floors are being requested. The Text field in the HUMAN-READABLE-INFO TLV is intended for human consumption.

The client may request the server to handle the floor request with a certain priority using a PRIORITY TLV.

12.2.1 Receiving a Response

After sending the FloorRequest message, the client follows the steps described in Section 12.1.1. Effectively, the client has issued a new floor request and asked the server to substitute an existing floor request with this new one. So, the FloorRequestStatus response to the new FloorRequest message will carry a different FLOOR-REQUEST-ID value than the old floor request.

If the modification is authorized by the server, the client will receive a FloorRequestStatus message with a Status Code 7 (Replaced) indicating that the old floor request has been replaced.

12.3 Requesting Information about Floor Requests

Clients request information about the current status of one or several floor request by sending a FloorRequestInfo message to the floor control server.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the client follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV). The client must insert a USER-ID TLV, which will be used by the server to authenticate and authorize the request.

If the client wants to request the status of a single floor request, it MUST insert a FLOOR-REQUEST-ID TLV that identifies the floor request at the server.

The client can also request information about all the ongoing floor requests associated with a particular user. In this case, the client MUST NOT insert a FLOOR-REQUEST-ID TLV. If the beneficiary of the floor requests the client is requesting information about is not the client issuing the FloorRequestInfo message (which is identified by the USER-ID TLV in the message) the client SHOULD insert a

BENEFICIARY-ID TLV.

12.3.1 Receiving a Response

On reception of the FloorRequestInfo message, the server will respond with a FloorRequestStatus message or with an error message. That is, the server will respond using the same message types as when it receives a FloorRequest message. Consequently, after sending the FloorRequestInfo message, the client follows the steps described in Section 12.1.1.

If the FloorRequestInfo message requested information about several floor requests, the FloorRequestStatus message will contain information about all of them.

12.4 Cancelling a Floor Request and Releasing a Floor

A client that wishes to cancel an ongoing floor request does so by sending a FloorRelease message to the floor control server. The ABNF in Section 6.2 describes the TLVs that a FloorRelease message can contain. In addition, the ABNF specifies which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the client follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV). The client must insert a USER-ID TLV, which will be used by the server to authenticate and authorize the request.

> Note that the FloorRelease message is also used to release a floor or floors that were granted to the client (from the protocol perspective both are ongoing floor requests). Using the same message in both situations helps resolve the race condition that occurs when the FloorRelease message and the FloorGrant message cross each other on the wire.

The client uses the FLOOR-REQUEST-ID that was received in the response to the FloorRequest message that the FloorRelease message is cancelling.

12.4.1 Receiving a Response

On reception of the FloorRelease message, the server will respond as with a FloorRequestStatus message or with an error message. That is, the server will respond using the same message types as when it receives a FloorRequest message. Consequently, after sending the

FloorRelease message, the client follows the steps described in Section 12.1.1.

It is possible that the FloorRelease message crosses on the wire with a FloorRequestStatus message from the server with a Request Status different from Cancelled. In any case, such a FloorRequestStatus message will not be a response to the FloorRelease message, because its Transaction ID will not match that of the FloorRelease.

12.5 Requesting Information about Floors

Clients request information about the status of one or several floors by sending a FloorInfo message to the floor control server.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the client follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV). The client must insert a USER-ID TLV, which will be used by the server to authenticate and authorize the request.

The client inserts in the message all the FLOOR-IDs it wants to receive information about. The server will send periodic information about all these floors. If the client does not want to receive information about a particular floor any longer, it sends a new FloorInfo message removing the FLOOR-ID of this floor. If the client does not want to receive information about any floor, it sends a FloorInfo message with no FLOOR-ID TLV.

12.5.1 Receiving a Response

A message from the server is considered to be a response to the FloorInfo message if the message from the server has the same Conference ID and Transaction ID as the FloorRequest message, as described in Section 10.1.

On reception of the FloorInfo message, the server will respond with a FloorStatus message or with an Error message. If the response is a FloorStatus message, it will contain information about one of the floors the client requested information about. If the client did not include any FLOOR-ID in its FloorInfo message, the FloorStatus message form the sever will not include any either.

After the first FloorStatus, the server will continue sending FloorStatus messages periodically informing the client about changes on the floors the client requested information about.

12.6 Checking the Liveness of a Server

A client that wishes to check the liveness of a server does so by sending a Ping message to the floor control server. The ABNF in Section 6.9 describes the TLVs that a Ping message can contain. In addition, the ABNF specifies which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the client follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV).

12.6.1 Receiving Responses

A message from the server is considered a response to the Ping message by the client if the message from the server has the same Conference ID and Transaction ID as the Ping message, as described in Section 10.1.

If the response is a PingAck message, the server is alive and the user identified by the User ID was authenticated successfully.

If the response is an Error message, the server could not process the Ping message for some reason, which is described in the Error message.

13. Chair Operations

This section specifies how clients acting as chairs can perform different operations, such as instructing the floor server to grant or revoke a floor, using the protocol elements described in earlier sections.

13.1 Obtaining Information about Floor Requests

A chair can obtain information about the floor requests that the floor control server receives in different ways, which include using out-of-band mechanisms. Chairs using BFCP to obtain such information use the procedures described in Section 12.5. As a result, they receive information about floor requests that relate to specific floors in FloorStatus messages from the floor control server.

13.2 Instructing the Floor Control Server

Chairs that wish to send instructions to a floor control server does so by sending a ChairAction message. The ABNF in Section 6.7

describes the TLVs that a ChairAction message can contain. In addition, the ABNF specifies which of these TLVs are mandatory, and which ones are optional.

The chair sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 10.1. Additionally, the chair follows the rules in Section 11.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the INTEGRITY TLV). The client must insert a USER-ID TLV, which will be used by the server to authenticate and authorize the request.

The ChairAction message contains instructions that apply to one or more floors within a particular floor request. The floor or floors are identified by FLOOR-ID TLVs and the floor request is identified by a FLOOR-REQUEST-ID TLV, which are carries in the ChairAction message.

> For example, if a floor request consists of two floors that depend on different chairs, each chair will grant its floor within the floor request. Once both chairs have grant their floor, the floor control server will grant the floor request as a whole. On the other hand, if one of the chairs denies its floor, the floor control server will deny the floor request as a whole, regardless of the other chair's decision.

The chair provides the new status for one or more floors within the floor request using a REQUEST-STATUS TLV. If the new status of the floor request is Accepted, the chair MAY use the Queue Position field to provide a queue position for the floor request. If the chair does not wish to provide a queue position, all the bits of the Queue Position field SHOULD be set to zero. The chair SHOULD use the Status Revoked to revoke a floor that was granted (i.e., Granted status) and to reject floor requests in any other status (e.g., Pending and Accepted).

> Note a floor request may involve several floors and that a ChairAction message could only deal with a subset of these floors (e.g., if a single chair is not authorized to manage all the floors). In this case, the REQUEST-STATUS that the chair provides in the ChairAction message might not be the actual status that the floor request gets at the server. The floor control server will combine the instructions received from the different chairs to come up with the actual status of the floor request.

The chair may use a HUMAN-READABLE-INFO TLV to state the reason why the floor or floors are being accepted, granted, or revoked. The Text in the HUMAN-READABLE-INFO TLV is intended for human consumption.

13.2.1 Receiving a Response

A message from the server is considered to be a response to the ChairAction message if the message from the server has the same Conference ID and Transaction ID as the ChairAction message, as described in Section 10.1.

A ChairActionAck message from the server confirms that the server has accepted the ChairAction message. An Error message indicates that the server could not process the ChairAction message for some reason, which is described in the Error message.

14. Server Operations

This section specifies how servers can perform different operations, such as granting a floor, using the protocol elements described in earlier sections.

On reception of a message from a client, the floor control server MUST check whether or not the value of the Conference ID matched an existing conference. If it does not, the floor server SHOULD send an Error message, as described in Section 14.7, with Error code 0 (Conference does not Exist).

On reception of a message from a client, the floor control server follows the rules in Section 11.2, which relate to the authentication of the message.

On reception of a message from a client, the floor control server MUST check whether or not it understands all the mandatory ('M' bit set) TLVs in the message. If the server does not understand all of them, the floor server SHOULD send an Error message, as described in Section 14.7, with Error code 2 (Authentication Failed). The Error message SHOULD list the TLVs that were not understood.

OPEN ISSUE: can servers use new new mandatory TLVs in their messages? Right now we do not have a way for clients to complain about unsupported TLVs received from the server.

14.1 Reception of a FloorRequest Message

The processing of a FloorRequest message by a server involves generating a FloorRequestStatus message. The floor control server SHOULD generate a FloorRequestStatus message as soon as possible. If the floor server cannot accept, grant, or deny the floor request right away (e.g., a decision from a chair is needed), it SHOULD use a Request Status value of Pending.

The server copies the Conference ID and the TRANSACTION-ID TLV from the FloorRequest into the FloorRequestStatus, as described in Section 10.2. Additionally, the server MUST assign an identitifier that is unique within the conference to this floor request, and insert it in a FLOOR-REQUEST-ID TLV into the FloorRequestStatus message. This indentifier will be used by the client to refer to this specific floor request in the future.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

At later time, when the status of the floor request changes, the floor control server SHOULD send a new FloorRequestStatus message with the appropriate Request Status. This FloorRequestStatus message MUST contain a FLOOR-REQUEST-ID TLV identifying the floor request, but MUST NOT contain any TRANSACTION-ID TLV. The server may add a HUMAN-READABLE-INFO TLV to provide extra information to the user about its decision.

> The rate at which the floor control server sends FloorRequestStatus messages is a matter of local policy. A server may choose to send a new FloorRequestStatus message every time the floor request moves in the floor request queue while another may choose to only send a new FloorRequestStatus message when the floor request is Granted or Denied.

Clients and chairs that request so need to be informed about changes in the status of a floor (e.g., a new floor request arrives). To accomplish this, the floor control server follows the rules in Section 14.4.

14.1.1 Floor Request Modification

If the FloorRequest message contains a FLOOR-REQUEST-ID TLV, it means that the client is requesting a floor control to be modified. In addition to the actions described in Section 14.1, if the modification is authorized, the floor control server SHOULD send a FloorRequestStatus message with Request Status 7 (Replaced) for the original floor request.

If the FloorRequest message contains an invalid FLOOR-REQUEST-ID, the floor control server SHOULD respond with an Error response with Error Code 3 (Floor Request ID Does Not Exist).

14.2 Reception of a FloorRequestInfo Message

The processing of a FloorRequestInfo message by a server involves generating a FloorRequestStatus message. The FloorRequestInfo message

can apply to a single floor request, identified by a FLOOR-REQUEST-ID, or to all the floor requests from a given user, the FloorRequestInfo does not carry a FLOOR-REQUEST-ID. The user is identified by a BENEFICIARY-ID TLV, or if this is not present, by a USER-ID TLV.

If the FloorRequestInfo message contains an invalid FLOOR-REQUEST-ID, the floor control server SHOULD respond with an Error response with Error Code 3 (Floor Request ID Does Not Exist).

On reception of a FloorRequestInfo message, the floor control server SHOULD generate a FloorRequestStatus message as soon as possible.

The server copies the Conference ID and the TRANSACTION-ID TLV from the FloorRequestInfo message into the FloorRequestStatus, as described in Section 10.2.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

14.3 Reception of a FloorRelease Message

The processing of a FloorRelease message by a server involves generating a FloorRequestStatus message. The FloorRelease message identifies the floor request it applies to using a FLOOR-REQUEST-ID. If the FloorRelease message contains an invalid FLOOR-REQUEST-ID, the floor control server SHOULD respond with an Error response with Error Code 3 (Floor Request ID Does Not Exist).

On reception of a FloorRelease message with a valid FLOOR-REQUEST-ID, the floor control server SHOULD generate a FloorRequestStatus message as soon as possible. If the floor server can authorize the release operation right away, it SHOULD use a Request Status value of Released, if the floor had been previously granted, or of Cancelled, if it had not. The server may add a HUMAN-READABLE-INFO TLV to provide extra information to the user about its decision.

The server copies the Conference ID and the TRANSACTION-ID TLV from the FloorRelease into the FloorRequestStatus, as described in Section 10.2.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

14.4 Reception of a FloorInfo Message

A server receiving a FloorInfo message from a client SHOULD keep this client informed about the status of the floors identified by

FLOOR-IDs in the FloorInfo message. Servers keep clients informed by using FloorStatus messages.

The information FloorInfo messages carry may depend on the user requesting the information. For example, a chair may be able to receive information about pending requests while a regular user may not be authorized to do so.

The server may provide information about a number of floor requests in a single FloorInfo message. For each floor request, the server provides its FLOOR-REQUEST-ID and its REQUEST-STATUS, and may provide further information such as its BENEFICIARY-ID.

On reception of a FloorInfo message with one or more FLOOR-ID TLVs, the server generates a FloorStatus message for one of the floors identified in the FloorInfo message.

The server copies the Conference ID and the TRANSACTION-ID TLV from the FloorInfo message into the FloorStatus message, as described in Section 10.2.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

After sending this message, the server SHOULD continue sending FloorStatus messages about all the floors the client requested information about. These messages MUST NOT carry a TRANSACTION-ID TLV.

> The rate at which the floor control server sends FloorStatus messages is a matter of local policy. A server may choose to send a new FloorRequestStatus message every time a new floor request arrives while another may choose to only send a new FloorStatus message when a new floor request is Granted.

14.5 Reception of a ChairAction Message

On reception of a ChairAction message, the floor control server checks whether the chair that send the message is authorized to perform the operation being requested. If the chair is authorized, the floor control server performs the operation and sends a ChairActionAck message to the client. If the new Status in the ChairAction message is Accepted and all the bits of the Queue Position field are zero, the server assigns a queue position (e.g., the last in the queue) to the floor request based on local policy (in case the server implements a queue).

The server copies the Conference ID and the TRANSACTION-ID TLV from

the ChairAction message into the ChairActionAck message, as described in Section 10.2.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

If the floor control server cannot find a floor request that matches the FLOOR-REQUEST-ID TLV in the ChairAction message the floor control server generates an Error message, as described in Section 14.7, with an Error code with a value of 3 (Floor Request ID Does Not Exist).

If the chair is not authorized to perform the operation being requested, the floor control server generates an Error message, as described in Section 14.7, with an Error code with a value of 4 (Unauthorized Operation).

14.6 Reception of a Ping Message

The processing of a Ping message by a server involves generating a PingAck message. On reception of a Ping message, the floor control server SHOULD generate a PingAck message as soon as possible. The server copies the Conference ID and the TRANSACTION-ID TLV from the Ping into the PingAck, as described in Section 10.2.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

14.7 Error Message Generation

Error messages are always sent in response to a previous message from the client as part of a client-initiated transaction. The ABNF in Section 6.11 describes the TLVs that an Error message can contain. In addition, the ABNF specifies which of these TLVs are mandatory, and which ones are optional.

Servers copy the Conference ID and the TRANSACTION-ID TLV from the message from the client into the Error message, as described in Section 10.2.

The server follows the rules in Section 11.2 which relate to authentication and the use of the INTEGRITY TLV.

15. BFCP and the Offer/Answer Model

The way a client obtains a the information needed to contact a BFCP floor control server is outside the scope of BCFP. Nevertheless, this section describes how to obtain such an information using an SDP [5] offer/answer [11] exchange.

User agents typically use the offer/answer model to establish a number of media streams of different types. Following this model, a BFCP connection is described as any other media stream by using an SDP m line.

15.1 Fields in the m Line

According to RFC 2327 [5], the m-line format is the following:

```
m=<media> <port> <transport> <fmt list>
```

The media field MUST have a value of "application". The port field is not used by BFCP, and MAY be set to any value chosen by the endpoint. A port field value of zero has the standard SDP meaning (i.e., rejection of the media stream).

The port field is set following the rules in [14]. Depending on the value of the setup attribute (disccused in Section 15.4), the port field contains the port the remote endpoint will initiate its TCP connection to, or is irrelevant (i.e., the endpoint will initiate the connection towards the remote endpoint) and should be set to a value of 9, which is the discard port. Since BFCP only runs on top of TCP, the port is always a TCP port.

We define two new values for the transport field: TCP/BFCP and TCP/TLS/BFCP.

The fmt (format) list is ignored for BFCP. The fmt list of BFCP m lines SHOULD contain a single "*" character.

The following is an example of an m line for a BFCP connection:

```
m=application 20000 TCP/BFCP *
```

15.2 The confid and userid SDP Parameters

We define the confid and the userid SDP media-level attributes. Their syntax is:

```
confid-attribute      = "a=confid: " conference-id
conference-id         = token

userid-attribute      = "a=userid: " user-id
user-id               = token
```

The confid and the userid attributes carry the integer representation

of a conference ID and a user ID respectively.

Endpoints which use the offer/answer model to establish BFCP connections MUST support the confid and the userid attributes. A floor control server acting as an offerer or as an answerers SHOULD include these attributes in its session descriptions.

15.3 The k line

If the offer/answer exchange is encrypted and integrity protected, the offerer MAY use an SDP k line to provide the answerer with a shared secret to be used to calculate the value of the INTEGRITY TLVs. The following is an example of a k line:

```
k=base64:c2hhcmVkLXNlY3JldA==
```

15.4 TCP Connection Management

The management of the TCP connection used to transport BFCP is performed using the setup and connid attributes as defined in [14].

The setup attribute indicates which of the endpoints (client or floor control server) initiates the TCP connection. The connid attribute handles TCP connection reestablishment.

15.5 Association between Streams and Floors

EDITOR'S NOTE: We need a way for clients that don't support CPCP to at a minimum learn enough information about floors to use the floor control protocol. This section will need to be harmonized with the media policy work.

We define the floorid SDP media-level attribute. Its syntax is:

```
floor-id-attribute    = "a=floorid:" token " mstream:" 1*(token)
```

The floorid attribute is used in BFCP m lines and associates a floor ID with a media stream. The token representing the floor ID is the integer representation of the 16-bit floorid to be used in BFCP. The token representing the media stream is a pointer to the media stream, which is identified by an SDP label attribute [draft-levin-mmmusic-sdp-media-label-00.txt].

Endpoints which use the offer/answer model to establish BFCP connections MUST support the floorid and the label attributes. A floor control server acting as an offerer or as an answerers SHOULD include these attributes in its session descriptions.

15.6 Example

The following is an example of an offer sent by a conference server to a user. For the purpose of brevity, the main portion of the session description is omitted in the examples, which only show m= lines and their attributes.

```
m=application 20000 TCP/BFCP *
k=base64:c2hhcmVkLXNlY3JldA==
a=setup:passive
a=connid:1
a=confid:4321
a=userid:1234
a=floorid:1 m-stream:10
a=floorid:2 m-stream:11
m=audio 20000 RTP/AVP 0
a=label:10
m=video 30000 RTP/AVP 31
a=label:11
```

The following is the answer returned by the user.

```
m=application 9 TCP/BFCP *
a=setup:active
a=connid:1
m=audio 25000 RTP/AVP 0
m=video 35000 RTP/AVP 31
```

16. Security Considerations

TBD.

17. IANA Considerations

TBD

17.1 SDP Attributes Registration

TBD:

18. Acknowledgments

The XCON WG chairs, Adam Roach and Alan Johnston, provided useful ideas for this document. Additionally, Xiaotao Wu, Paul Kyzivat, Jonathan Rosenberg, and Miguel A. Garcia-Martin provided useful comments.

19. References

19.1 Normative References

[1] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

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

[3] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[4] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[5] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[6] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[7] Petke, R. and I. King, "Registration Procedures for URL Scheme Names", BCP 35, RFC 2717, November 1999.

[8] Hinden, R., Carpenter, B. and L. Masinter, "Format for Literal IPv6 Addresses in URL's", RFC 2732, December 1999.

[9] Gulbrandsen, A., Vixie, P. and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[10] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[11] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[12] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part

Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[13] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[14] Yon, D., "Connection-Oriented Media Transport in SDP", draft-ietf-mmusic-sdp-comedia-05 (work in progress), March 2003.

19.2 Informational References

[15] Schulzrinne, H., "Requirements for Floor Control Protocol", draft-ietf-xcon-floor-control-req-00 (work in progress), January 2004.

[16] Koskelainen, P. and H. Khartabil, "An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Usage for Conference Policy Manipulation", draft-koskelainen-xcon-xcap-cpcp-usage-02 (work in progress), February 2004.

[17] Rosenberg, J. and H. Schulzrinne, "A Session Initiation Protocol (SIP) Event Package for Conference State", draft-ietf-sipping-conference-package-03 (work in progress), February 2004.

[18] Arkko, J., "MIKEY: Multimedia Internet KEYing", draft-ietf-msec-mikey-08 (work in progress), December 2003.

[19] Rosenberg, J., "A Session Initiation Protocol (SIP) Event Package for Modification Events for the Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Managed Documents", draft-ietf-simple-xcap-package-01 (work in progress), February 2004.

Authors' Addresses

Gonzalo Camarillo
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: Gonzalo.Camarillo@ericsson.com

Joerg Ott
Universitaet Bremen
MZH 5180
Bibliothekstr. 1
Bremen D-28359
Germany

EMail: jo@tzi.org

Keith Drage
Lucent Technologies
Windmill Hill Business Park
Swindon
Wiltshire SN5 6PP
UK

EMail: drage@lucent.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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