Internet Engineering Task Force
INTERNET-DRAFT
draft-ietf-rmt-bb-lct-01.ps

RMT WG
M.Luby/Digital Fountain
J.Gemmell/Microsoft
L.Vicisano/Cisco
L.Rizzo/ACIRI and Univ. Pisa
M.Handley/ACIRI
J. Crowcroft/UCL
19 July 2001
Expires: January 2002

Layered Coding Transport: A massively scalable content delivery transport

Status of this Document

Abstract

This document describes Layered Coding Transport, a massively scalable reliable content delivery and stream delivery transport, hereafter referred to as LCT. LCT can be used for multi-rate delivery to large sets of receivers. In LCT, scalability and congestion control are supported through the use of layered coding techniques. Coding techniques can be combined with LCT to provide support for reliability.

Congestion control is receiver driven, and is achieved by sending packets in the session to multiple “LCT channels”, and having receivers join and leave LCT channels (thus adjusting their reception rate) in reaction to network conditions in a manner that is network friendly.

Table of Contents

1. Introduction

This document describes a massively scalable reliable content delivery and stream delivery transport, Layered Coding Transport (LCT), for multi-rate content delivery primarily designed to be used with the IP multicast network service, but MAY also be used with other basic underlying network or transport services such as unicast UDP. LCT supports both reliable and unreliable delivery, and supports congestion control mechanisms which conform to RFC2357.

LCT is a building block as defined in RFC3048. Complete protocol instantiations MAY be built by combining the LCT framework with other components. A complete protocol instantiation that uses LCT MUST include a congestion control protocol that is compatible with LCT and that conforms to RFC2357. A complete protocol instantiation that uses LCT MAY include a scalable reliability protocol that is compatible with LCT, it MAY include an session control protocol that is compatible with LCT, and it MAY include other protocols such as security protocols.

LCT is presumed to be used with an underlying network or transport service that is a "best effort" service that does not guarantee packet reception, packet reception order, and which does not have any support for flow or congestion control. For example, the Any-Source Multicast (ASM) model of IP multicast as defined in RFC1112 is such a "best effort" network service. While the basic service provided by RFC1112 is largely scalable, providing congestion control or reliability should be done carefully to avoid severe scalability limitations, especially in presence of heterogeneous sets of receivers.

Scalability refers to the behavior of the protocol in relation to the number of receivers and network paths, their heterogeneity, and the ability to accommodate dynamically variable sets of receivers. Scalability limitations can come from memory or processing requirements, or from the amount of packet traffic generated by the protocol. In turn, such limitations may be a consequence of the features that a complete reliable content delivery or stream delivery protocol is expected to provide.

Congestion control refers to the ability of the protocol to adapt its throughput to the available bandwidth on the path to the receivers, and to share bandwidth fairly with competing flows such as TCP. It is REQUIRED that protocols implement some form of congestion control on each session so that they not compete unfairly with existing and adaptive protocols such as TCP.

For multi-rate protocols, the sender typically sends packets at a constant aggregate rate to multiple channels, but individual receivers adjust which portion of these packets they attempt to receive by adjusting which set of channels they are joined to at each point in time depending on the available bandwidth between the receiver and the sender, but independent of all other receivers. Thus, for multi-rate protocols, the packet reception rate of each receiver may vary dynamically according to the available bandwidth between each receiver and the sender, independent of the other receivers. For single-rate protocols, the sender typically sends packets to one channel at variable rates over time depending on feedback from receivers, and all receivers attempt to receive all packets transmitted by the sender at all points in time. Thus, for single-rate protocols, the packet reception rate of all receivers may vary dynamically over time in exactly the same way dependent on the feedback of other receivers to the sender. Generally, a multi-rate protocol is preferable to a single-rate protocol in a heterogeneous receiver environment, but only if it can be achieved without sacrificing scalability.

Layered coding refers to the ability to produce a coded stream that can be split into multiple layers that can be sent to different channels. The coded stream can be generated either from a fixed piece of content, or from an ongoing data stream, and has the property that the quality experienced by a receiver (in terms of quality of playout, or overall transfer speed) is proportional to how many of the layers the receiver is joined to. LCT MAY be combined with a reliability protocol such as the general class of protocols described in [7]. LCT MUST be combined with a congestion control protocol that is compliant with RFC2357, and this MAY be either single-rate or multi-rate congestion control over the entire session. It is most compelling to use LCT in conjunction with a layered congestion control protocol such as the class of protocols described in [9], and specified in [9], which are multi-rate congestion control protocols.

The concept of layered coding was first introduced with reference to audio and video streams. For example, the information associated with a TV broadcast can be partitioned into three layers, corresponding to black and white, color, and HDTV quality. Receivers can experience different quality without the need for the sender to replicate information in the different layers.

The concept of layered coding can be naturally extended to reliable content delivery protocols when Forward Error Correction (FEC) techniques are used for coding the data stream [12] [10] [3] [14] [15] [1]. By using FEC, the data stream is transformed in such a way that reconstruction of a data object does not depend on the reception of specific data packets, but only on the number of different packets received. As a result, by increasing the number of layers a receiver is receiving from, the receiver can reduce the transfer time accordingly. More details on the use of FEC for reliable content delivery can be found in [6]. Reliable protocols aim at giving guarantees on the reliable delivery of data from the sender to the intended recipients. Guarantees vary from simple packet data integrity to reliable delivery of a precise copy of a data object to all intended recipients. Several reliable content delivery protocols have been built on top of IP multicast, but scalability was not a design goal for many of them. In some cases, scalability is achieved by introducing changes to routers or other infrastructure (see for example [13]), an approach which has an impact on near term deployment across the Internet.

Two of the key difficulties in scaling reliable content delivery using IP multicast are dealing with the amount of data that flows from receivers back to the sender, and the associated response (generally data retransmissions) from the sender. Protocols that avoid any such feedback, and minimize the amount of retransmissions, can be massively scalable. LCT relies on the availability of FEC codes or a layered codec to achieve reliability with little or no feedback.

In this document we present the architecture and abstract LCT packet structure, and illustrate its support for multi-rate layered congestion control.

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

1.1. Related Documents

A more in-depth description of the use of FEC in Reliable Multicast Transport (RMT) protocols is given in [6]. Some of the FEC codecs that MAY be used by LCT for reliable content delivery are specified in [7]. LCT reserves opaque header fields that can be used to transport information related to the payload encoding.

Implementors of LCT MUST also implement congestion control in accordance to RFC2357, where the congestion control is over the entire session. Some possible schemes are specified in

[9]. LCT reserves opaque header fields that can be used by the congestion control scheme to transport information related to congestion control.

It is RECOMMENDED that LCT implementors use some authentication scheme to protect the protocol from attacks. An example of a possibly suitable scheme is described in [11].

1.2. Environmental Requirements and Considerations

LCT is intended for congestion controlled, multi-rate delivery of objects and streams (both reliable content delivery and streaming of multimedia information).

LCT is most applicable for delivery of objects or streams of substantial length, i.e., objects or streams that range in length from hundreds of kilobytes to many gigabytes, and whose transfer time is in the order of tens of seconds or more.

LCT is directly applicable to all multicast enabled networks, including asymmetric networks, wireless networks, and satellite networks. Thus, the inherent raw scalability of LCT is unlimited. However, when other specific applications are built on top of LCT, then these applications by their very nature may limit scalability. For example, if an application requires receivers to retrieve out of band information in order to join a session, or an application allows receivers to send requests back to the sender in order to extend an ongoing session, then the scalability of the application is limited by the ability to send, receive, and process this additional data.

LCT requires that the underlying network or transport layer can deliver and demultiplex LCT packets for a given LCT session, and supply packet length information to the LCT receiver. In IP networks, this is often achieved by using UDP, or any protocol that can provide an equivalent service, as the underlying transport protocol.

In this document, we refer to the original multicast service model defined in RFC1112 as Any-Source Multicast, or ASM for short. We refer to the multicast service model defined in [5] as Source-Specific Multicast, or SSM for short. LCT does not require reverse multicast connectivity, i.e. LCT receivers do not send multicast traffic. As such, LCT works with both ASM and SSM.

The definition of an LCT channel used throughout this document is slightly different with ASM and with SSM. When using ASM, a sender S sends LCT packets to a multicast group G, and the LCT channel address consists of the pair (S,G), where S is the IP address of the sender and G is a multicast group address. When using SSM, a sender S sends LCT packets to an SSM channel (S,G), and the LCT channel address coincides with the SSM channel address.

SSM is more attractive to deployment of LCT than ASM for several reasons. First, a sender may allocate and use several LCT channels in a session, sessions may come and go dynamically. A sender can locally allocate unique SSM channel addresses, and this makes allocation of LCT channel addresses easy with SSM. To allocate LCT channel addresses using ASM, the sender must uniquely chose the ASM multicast group address across the scope of the group, and this makes allocation of LCT channel addresses more difficult with ASM.

In general SSM performs well even in presence of very large and dynamically changing receiver sets. Changes in the multicast tree topology with SSM are light weight operations (a new branch from the receiver towards S grows when a receiver joins, and the branch is deleted when the receiver leaves), whereas with ASM changes can be heavier weight (involving transitions from a

(*,G)-tree rooted at an RP to the tree rooted at S). This efficiency is important when a congestion control protocol is used with LCT that relies upon joining and leaving channels to nimbly increase and decrease reception rate.

LCT channels and SSM channels coincide, and thus the receiver will only receive packets sent to the requested LCT channel. With ASM, the receiver joins an LCT channel by joining a multicast group G, and all packets sent to G, regardless of the sender, may be received by the receiver. In either case, receivers should use mechanisms to filter out packets from unwanted sources. Thus, SSM has compelling security advantages over ASM for prevention of denial of service attacks.

LCT also requires receivers to obtain Session Description Information before joining a session, as described in Section 4.1. The session description could be in a form such as SDP as defined in RFC2327, or XML metadata, or HTTP/Mime headers as defined in RFC2068, and distributed with SAP [4], using RCCP [8], HTTP, or in other ways.

The particular layered encoder and congestion control protocols used with LCT have an impact on the performance and applicability of LCT. For example, some layered encoders used for video and audio streams can produce a very limited number of substreams, thus providing a very coarse control in the reception rate of packets by receivers in a session. When LCT is used for reliable data transfer, some FEC codecs are inherently limited in the size of the object they can encode, and for objects larger than this size the reception overhead on the receivers can grow substantially.

As another example, some networks are not amenable to some congestion control protocols that could be used with LCT. In particular, for a satellite or wireless network, there may be no mechanism for receivers to effectively reduce their reception rate since there may be a fixed transmission rate allocated to the session.

2. General Architecture

An LCT session comprises all LCT packets sent to one or more LCT channels from a single sender, and pertaining to the transmission of one or more objects that can be of interest for the receivers.

For example, an LCT session could be used to deliver a TV channel using three channels. Receiving the first channel could allow black and white reception, receiving the first two channels would permit color reception, whereas receiving the set of three channels delivers HDTV quality images. Objects in this example could correspond to individual programs (movies, news, commercial) being transmitted.

As another example, a reliable LCT session could be used to reliably deliver hourly-updated weather maps (objects) using ten LCT channels at different rates, using FEC coding. A receiver MAY join and concurrently receive LCT packets from subsets of these channels, until it has enough packets in total to recover the object, then leave the session (or remain there listening to control information only) until it is time to receive the next object. In this case, the quality metric is the time required to receive each object.

Before joining a session, the receivers MUST obtain a session description, which MUST include the relevant session parameters needed by a receiver to participate in the session. The session description is determined and agreed upon by the senders, and typically communicated to the receivers out of band. In some cases, part of the session description MAY be included in the LCT packet.

An encoder MAY be used to generate the data that is placed in the LCT packet payload in order to provide reliability. A suitable decoder is used to reproduce the original information from the payload. There MAY be a reliability packet header that follows the LCT packet header if such an encoder and decoder is used. The reliability packet header helps to describe the encoding data carried in the payload of the packet. The format of the reliability packet header depends on the coding used, and this is negotiated out-of-band. As an example, one of the FEC packet headers described in [7] could be used.

For LCT, when layered congestion control is used, congestion control is achieved by sending LCT packets associated with a given session to several LCT channels. Individual receivers dynamically join one or more of these channels, according to the network congestion as seen by the receiver. LCT packet headers include an opaque field which MUST be used to convey congestion control information to the receivers. The actual congestion control scheme to use with LCT is negotiated out-of-band. Some examples of congestion control protocols that MAY be suitable for content delivery are described in [9]. Other congestion controls MAY be suitable when LCT is used for a streaming application.

LCT can be used with other congestion control protocols such as the one described in [14], or router-assisted schemes where the selection of which packets to forward is performed by routers. This latter approach potentially allows for finer grain congestion control and a faster reaction to network congestion, but requires changes to the router infrastructure. See [2] for a preliminary design description. We do not discuss this approach further in this document.

2.1. Delivery service models

LCT can support several different delivery service models. Two examples are briefly described here.

Push service model.

One way a push service model can be used for reliable content delivery is to deliver a series of objects. For example, a receiver could join the session and dynamically adapt the number of LCT channels the receiver is joined to until enough packets have been received to reconstruct an object. After reconstructing the object the receiver may stay in the session and wait for the transmission of the next object.

The push model is particularly attractive in satellite networks and wireless networks. In these cases, a session may consist of one fixed rate LCT channel.

On-demand content delivery model.

For an on-demand content delivery service model, senders typically transmit for some given time period selected to be long enough to allow all the intended receivers to join the session and recover the object. For example a popular software update might be transmitted using LCT for several days, even though a receiver may be able to complete the download in one hour total of connection time, perhaps spread over several intervals of time.

In this case the receivers join the session, and dynamically adapt the number of LCT channels they subscribe to (and the reception quality) according to the available bandwidth. Receivers then drop from the session when they have received enough packets to recover the object.

As an example, assume that an object is 50 MB. The sender could set the data rate on the lowest LCT channel to 50 1KB packets per second, so that receivers using just this LCT channel could complete reception of the object in 1,000 seconds in absence of loss, and would be able to complete reception even in presence of some substantial amount of losses with the use of coding for reliability. Furthermore, the sender could use a number of LCT channels such that the aggregate data rate when using all LCT channels is 1,000 1KB packets per second, so that a receiver could be able to complete reception of the object in as little 50 seconds (assuming no loss and that the congestion control mechanism immediately converges to the use of all LCT channels).

Other service models.

There are many other delivery service models that LCT can be used for that are not covered above. As examples, a live streaming or an on-demand archival content streaming service model. The description of the many potential applications, the appropriate delivery service model, and the additional mechanisms to support such functionalities when combined with LCT is beyond the scope of this document. This document only attempts to describe the minimal common scalable elements to these diverse applications using LCT as the delivery transport.

2.2. Congestion Control

The specific congestion control protocol to be used for LCT sessions depends on the type of content to be delivered. While the general behavior of the congestion control protocol is to reduce the throughput in presence of congestion and gradually increase it in the absence of congestion, the actual dynamic behavior (e.g. response to single losses) can vary.

Some possible congestion control protocols for reliable content delivery using LCT are described in [9]. Different delivery service models might require a different congestion control protocols.

3. LCT packets

The type of packet used by LCT sessions is "LCT Packet". LCT packets are sent by senders to LCT channels.

Some instances of LCT sessions MAY require the generation of feedback from the receivers to the sender. However, the mechanism for doing this is outside the scope of LCT.

The LCT packet format described in this document is intended to be used in conjunction with the UDP transport protocol as defined in RFC768 or other transport protocols that satisfy the requirements stated in Section 1.2, specifically about demultiplexing and delivery of packet size information.

LCT packets consist of an LCT packet header and an OPTIONAL LCT packet payload, as shown in Figure 1. When present, the LCT packet payload immediately follows the LCT packet header.

The LCT packet payload MAY contain headers for other protocols, such as reliability protocols.

LCT packet headers have variable size, which is specified by a length field in the third byte of the header. In the LCT packet header, all integer fields are carried in "big-endian" or "network order" format, that is, most significant byte (octet) first. Bits designated as "padding" or "reserved" (r) MUST by set to 0 by senders and ignored by receivers. Unless otherwise noted, numeric constants in this specification are in decimal (base 10).

3.1. LCT packet format

The format of LCT packets is depicted in Figure 1.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   V   |     r     | I |S|E|X|A|B|    HDR_LEN    | Codepoint (CP)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Congestion Control Information (CCI)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Transport Object Identifier (TOI, if I >= 1)           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                TOI continued  (if I >= 2)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                TOI continued  (if I = 3)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                TOI continued  (if I = 3)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Sender Current Time (SCT, if S = 1)             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Expected Residual Time (ERT, if E = 1)           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Header Extensions (if X = 1 )                  |
|                                                               |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                                                               |
|                          Payload                              |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1 - LCT packet format

3.2. LCT packet header fields

The function each field in LCT packet headers is the following. Fields marked as "1" mean that the corresponding bits MUST be set to "1" by the generating agent. Fields marked as "r" or "0" mean that the corresponding bits MUST be set to "0" by the generating agent.

LCT version number (V): 4 bits

Indicates the LCT protocol version. The LCT version number for this specification is 0.

Reserved (r): 5 bits

Reserved for future use. A sender MUST set these bits to zero and a receiver MUST ignore these bits.

Transport Object Identifier flag (I): 2 bits

I=0 indicates no Transport Object Identifier (TOI) field is present.
I=1 indicates that a 32-bit TOI field is present.
I=2 indicates that a 64-bit TOI field is present.
I=3 indicates that a 128-bit TOI field is present.
The TOI field indicates which object within the session this LCT packet pertains to.

Sender Current Time present flag (S): 1 bit

S = 0 indicates that the Sender Current Time (SCT) field is not present.
S = 1 indicates that the SCT field is present.
The SCT is inserted by senders to indicate to receivers how long the session has been in progress.

Expected Residual Time present flag (E): 1 bit

E = 0 indicates that the Expected Residual Time (ERT) field is not present.
E = 1 indicates that the ERT field is present.
The ERT is inserted by senders to indicate to receivers how much longer the session will continue.
Senders MUST NOT set E = 1 when the ERT for the session is more than 2^32-1 time units (approximately 49 days), where time is measured in units of milliseconds.

Header extension present flag (X): 1 bit

X = 0 indicates no Header Extensions are present.
X = 1 indicates that Header Extensions are present.
Header Extensions are used in LCT to accommodate OPTIONAL header fields which are not always used or have variable size.

Close TOI flag (A): 1 bit

Normally, the Close TOI flag is set to 0. The sender MAY set the Close TOI flag to 1 when termination of transmission of LCT packets for the object identified by the TOI is imminent. The Close TOI flag MAY be set to 1 in just the last LCT packet transmitted for the object, or it MAY be set to 1 in the last couple of seconds LCT packets transmitted for the object. Once the sender sets the Close TOI flag to 1 in one packet for a particular object, the sender SHOULD set the Close TOI flag to 1 in all subsequent packets for the object until termination of transmission of LCT packets for the object. A received packet with the Close TOI flag set to 1 indicates to a receiver that the sender will immediately stop sending LCT packets for the object identified by the TOI. When a receiver receives a packet with the Close TOI flag set to 1 then it should assume that no more LCT packets will be sent to the session for this object.

Close Session flag (B): 1 bit

Normally, the Close Session flag is set to 0. The sender MAY set the Close Session flag to 1 when termination of transmission of LCT packets for the session is imminent. The Close Session flag MAY be set to 1 in just the last LCT packet transmitted for the session, or it MAY be set to 1 in the last couple of seconds LCT packets transmitted for the session. Once the sender sets the Close Session flag to 1 in one packet, the sender SHOULD set the Close Session flag to 1 in all subsequent packets until termination of transmission of LCT packets for the session. A received packet with the Close Session flag set to 1 indicates to a receiver that the sender will immediately stop sending LCT packets for the session. When a receiver receives a packet with the Close Session flag set to 1 then it should assume that no more LCT packets will be sent to the session.

LCT packet header length (HDR_LEN): 8 bits

Length of the LCT packet header in units of 32-bit words (excluding IP or UDP headers). This field can be used for direct access to the beginning of the LCT packet payload.

Codepoint (CP): 8 bits

An opaque identifier which is passed to the payload decoder to convey information on the codec being used for the payload. The mapping between the codepoint and the actual codec is defined on a per session basis and communicated out-of-band as part of the session description information. The use of the CP field is similar to the Payload Type (PT) field in RTP headers as described in RFC1889.

Congestion Control Information (CCI): 32 bits

Used to carry congestion control information, e.g. for the congestion control specified in [9] or other congestion control schemes. This field is opaque for the purpose of this specification.

Transport Object Identifier (TOI): 32, 64 or 128 bits (OPTIONAL)

This field indicates which object within the session this LCT packet pertains to. For example, a sender might send a number of files in the same session, using TOI=0 for the first file, TOI=1 for the second one, etc. The mapping of TOI field values to objects MUST be done out of band.
This field MUST NOT be present if I=0.
This field MUST be 32 bits if I=1.
This field MUST be 64 bits if I=2.
This field MUST be 128 bits if I=3.

Sender Current Time (SCT): 32 bits (OPTIONAL)

This field represents the current clock at the sender at the time this packet was transmitted, measured in units of 1ms and computed modulo 2^32 units from the start of the session. This field MUST NOT be present if S=0 and MUST be present if S=1.

Expected Residual Time (ERT): 32 bits (OPTIONAL)

This field represents the sender expected residual transmission time for the current session, measured in units of 1ms.

This field MUST NOT be present if E=0 and MUST be present if E=1.

3.3. Header-Extension Fields

To allow for additional header fields and to extend the size of some of the predefined fields, the LCT packet header contains an additional header field flag, "X". If "X" is set to 0 then no additional header fields are included within the LCT packet header beyond the predefined fields. When additional headers beyond the predefined fields are used, the value of "X" within the LCT packet header MUST be set to 1.

Examples of use of Header Extensions include:

- o Extended-size version of already existing header fields.
- o Sender and Receiver authentication information.

If present, Header Extensions MUST be processed to ensure that they are recognized before performing any congestion control procedure or otherwise accepting an LCT packet. LCT packets with unrecognised Header Extensions MUST be discarded by the receiving agent, hence the expected use of extensions SHOULD be signaled out-of-band before session startup.

There are two formats for Header Extension fields, as depicted below. The first format is used for variable-length extensions, with Header Extension Type (HET) values between 0 and 63. The second format is used for fixed length (one word) extension, using HET values from 64 to 127.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|L| HET (<=63)  |       HEL     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
.                                                               .
.              Header Extension Content (HEC)                   .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|L| HET (>=64)  |       Header Extension Content (HEC)          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5 - Format of additional headers

The explanation of each sub-field is the following.

Last Header Extension (L): 1 bit

MUST be set to 1 in the last Header Extension field present in an LCT packet header, MUST be set to 0 in all the others.

Header Extension Type (HET): 7 bits

The type of the Header Extension. This document defines a number of possible types. Additional types may be defined in future version of this specification. HET values from 0 to 63 are used for variable-length Header Extensions. HET values from 64 to 127 are used for fixed-length Header Extensions.

Header Extension Length (HEL): 8 bits

The length of the whole Header Extension field, expressed in multiples of 32-bit words. This field MUST be present for variable-length extensions (HET between 0 and 63) and MUST NOT be present for fixed-length extensions (HET between 64 and 127).

Header Extension Content (HEC): variable length

The content of the Header Extension. The format of this sub-field depends on the Header Extension type. For fixed-length Header Extensions, the HEC is 24 bits. For variable-length Header Extensions, the HEC field has variable size, as specified by the HEL field. Note that the length of each Header Extension field MUST be a multiple of 32 bits. Also note that the total size of the LCT packet header, including all Header Extensions and all OPTIONAL header fields, cannot exceed 255 32-bit words.

An LCT packet with Header Extensions MUST NOT have space between the end of the last Header Extension and the beginning of the LCT packet payload.

All senders and receivers of LCT packets MUST support the EXT_NOP Header Extension.

The following Header Extension types are defined:

EXT_NOP=0 No-Operation extension.
The information present in this extension field MUST be ignored by receivers.

EXT_CCI_V=1 Congestion Control Information extension, variable length.
This extension field extends the CCI field present in the fixed part of the header. It is used when the congestion control information does not fit in the 32-bit CCI field. When this option is present, the concatenation of the 32-bit CCI field in the fixed part of the header together with the variable length value provided in this option is used as the congestion control information. The interpretation of the data contained in EXT_CCI_V MUST be negotiated out-of-band. The use of EXT_CCI_V and EXT_CCI_F is mutually exclusive.

EXT_AUTH=2 Authentication extension
Information used to authenticate the sender of the LCT packet. If present, the format of this Header Extension and its processing MUST be communicated out-of-band as part of the session description.
It is RECOMMENDED that senders provide some form of authentication on the LCT packets they transmit. If EXT_AUTH is present, whatever authentication checks that can be performed immediately upon reception of the packet MUST be performed before accepting the packet and performing any congestion control-related action on it.

Some authentication schemes impose a delay of several seconds between when a packet is received and when the packet is fully authenticated. Any congestion control related action that is appropriate MUST NOT be delayed by any such full authentication delay.

EXT_CCI_F=65
Congestion Control Information extension, fixed length.
This extension field extends the CCI field present in the fixed part of the header. It is used when the congestion control information does not fit in the 32-bit CCI field. When this option is present, the concatenation of the 32-bit CCI field in the fixed part of the header together with the 24-bit fixed length value provided in this option is used as the congestion control information. The interpretation of the data contained in EXT_CCI_F MUST be negotiated out-of-band. The use of EXT_CCI_V and EXT_CCI_F is mutually exclusive.

4. Procedures

4.1. Sender Operation

Before a session starts, an LCT sender MUST make available all applicable information regarding the session. This information could include, but is not limited to:

- o The number of LCT channels;
- o The addresses, port numbers and data rates used for each LCT channel;
- o The format of the payload. For example, the payload could contain other protocol headers such as an FEC header. Then for example the information could include the mapping of codepoints used in the session to FEC codec types and parameters;
- o The congestion control scheme being used;
- o The possible Header Extensions that could be used in the session;
- o The mapping of TOI value(s) to objects for the session;
- o The authentication scheme being used, and all relevant information which is necessary for sender authentication purposes;

The session description could be in a form such as SDP as defined in RFC2327, XML metadata, HTTP/Mime headers, etc. It might be carried in a session announcement protocol such as SAP [4], obtained using RCCP session control as described in [8], located on a Web page with scheduling information, or conveyed via E-mail or other out of band methods. Discussion of session description format, and distribution of session descriptions is beyond the scope of this document.

Within an LCT session, an LCT sender transmits a sequence of LCT packets each containing a payload encoded according to one of the codecs defined in the session description. LCT packets are sent over one or more LCT channels which together constitute a session. Transmission rates MAY be different in different channels. This document does not specify the LCT packet payload, nor the order in which LCT packets are transmitted, nor the organization of a session into multiple channels. Although these issues affect the efficiency of the protocol, they do not affect the correctness nor the inter-operability between senders and receivers.

Multiple objects can be carried within the same LCT session. In this case, each object is identified by a unique TOI. Objects MAY be transmitted sequentially, or they MAY be transmitted concurrently.

Typically, the sender(s) continues to send LCT packets in a session until the transmission is considered complete. The transmission MAY be considered complete when some time has expired, a certain number of packets have been sent, or some out of band signal (possibly from a higher level protocol) has indicated completion by a sufficient number of receivers.

The specification of the processing of the payload carried in LCT packets is beyond the scope of this document. LCT will act as a transport layer and convey payload and associated information (Codepoint and TOI) to the receivers and any complete protocol using LCT will implement congestion control .

For the reasons mentioned above, this document does not pose any restriction on LCT packet sizes. However, network efficiency considerations recommend that the sender uses as large as possible payload size, but in such a way that LCT packets do not exceed the network's maximum transmission unit size (MTU), or fragmentation coupled with packet loss might introduce severe inefficiency in the transmission.

It is also RECOMMENDED that all LCT packets have the same or very similar sizes, as this can have a severe impact on the effectiveness of congestion control schemes such as the ones described in [9]. A sender of LCT packets MUST implement the sender-side part of one of the congestion control schemes that is in accordance with RFC2357, and the corresponding receiver congestion control scheme MUST be communicated out of band and implemented by any receivers participating in the session.

4.2. Receiver Operation

Receivers can operate differently depending on the delivery service model. For example, for an on demand service model receivers MAY join a session, obtain the necessary encoding symbols to reproduce the object, and then leave the session. As another example, for a streaming service model a receiver MAY be continuously joined to a set of LCT channels to download all objects in a session.

To be able to participate in a session, a receiver MUST first obtain the relevant session description information as listed in Section 4.1.

To be able to participate in a session, a receiver MUST implement the congestion control protocol specified in the session description. If a receiver is not able to implement the congestion control protocol used in the session, it MUST NOT join the session.

If sender authentication information is present in an LCT packet header, it MUST be used as specified in Section 3.3. If a receiver is unable to implement the authentication mechanism used by the session, it MUST NOT join the session.

To be able to be a receiver in a session, the receiver MUST be able to process the LCT packet header. The receiver MUST be able to discard, forward, store or process the LCT packet payload. If a receiver is not able to process a LCT packet header, it MUST either drop from the session, or reduce the receive bandwidth to the minimum value allowed by the congestion control protocol being used.

When the session is transmitted on multiple LCT channels, receivers MUST do it according to the specified startup behavior of the congestion control protocol itself. For a layered transmission on multiple channels, this typically means that a receiver will initially join only a minimal set of LCT channels, possibly a single one. This rule has the purpose of preventing receivers from starting at high data rates.

Multiple objects can be carried within the same LCT session. In this case, each object is identified by a unique TOI. Note that even if a server stops sending LCT packets for an old object before starting to transmit LCT packets for a new object, both the network and the underlying protocol layers can cause some reordering of packets, especially when sent over different LCT channels, and thus receivers MUST NOT assume that the reception of an LCT packet for a new object means that there are no more LCT packets in transit for the previous one, at least for some amount of time.

A receiver MAY be concurrently joined to multiple LCT sessions. The receiver MUST perform congestion control on each such LCT session. If the congestion control protocol allows the receiver some flexibility in terms of its actions within a session then the receiver MAY make choices to optimize the packet flow performance across the multiple LCT sessions, as long as the receiver still adheres to the congestion control rules for each LCT session individually.

5. Security Considerations

LCT can be subject to denial-of-service attacks by attackers which try to confuse the congestion control mechanism, or send forged packets to the session which would prevent successful reconstruction of large portions of the data stream.

The same exact problems are present in TCP, where an attacker can forge packets and either slow down or increase the throughput of the session, or replace parts of the data stream with forged data. If the stream is carrying compressed or otherwise coded data, even a single forged packet could also cause incorrect reconstruction of the rest of the data stream.

It is therefore RECOMMENDED that LCT agents implement some form of authentication to protect themselves against such attacks.

6. IANA Considerations

No information in this specification is subject to IANA registration.

Building blocks components used by LCT may introduce additional IANA considerations.

7. Intellectual Property Issues

No specific reliability protocol or congestion control protocol is specified or referenced as mandatory in this document.

LCT MAY be used with congestion control protocols and other protocols which are proprietary, or have pending or granted patents.

8. Acknowledgments

Thanks to Vincent Roca, Bruce Lueckenhoff and Hayder Radha for detailed comments on this document.

[1] Byers, J.W., Luby, M., Mitzenmacher, M., and Rege, A., "A Digital Fountain Approach to Reliable Distribution of Bulk Data", Proceedings ACM SIGCOMM '98, Vancouver, Canada, Sept 1998.

[2] Cain, B., Speakman, T., and Towsley, D., "Generic Router Assist (GRA) Building Block, Motivation and Architecture", Internet Draft draft-ietf-rmt-gra-arch-00.txt, a work in progress.

[3] Gemmell, J., Schooler, E., and Gray, J., "FCast Scalable Multicast File Distribution: Caching and Parameters Optimizations", Technical Report MSR-TR-99-14, Microsoft Research, Redmond, WA, April, 1999.

[4] Handley, M., "SAP: Session Announcement Protocol", Internet Draft, IETF MMUSIC Working Group, Nov 1996, a work in progress.

[5] Holbrook, H., Cain, B., "Source-Specific Multicast for IP", Internet Draft, SSM Working Group, March 2001, draft-holbrook-ssm-arch-02.txt, a work in progress.

[6] Luby, M., Gemmell, Vicisano, L., J., Rizzo, L., Handley, M., Crowcroft, J., "The use of Forward Error Correction in Reliable Multicast", Internet Draft draft-ietf-rmt-info-fec-00.txt, November 2000, a work in progress.

[7] Luby, M., Vicisano, L., Gemmell, J., Rizzo, L., Handley, M., Crowcroft, J., "RMT BB Forward Error Correction Codes", Internet Draft draft-ietf-rmt-bb-fec-03.txt, July 2001.

[8] Luby, M., Hernek, D., Kushi, D., Rasmussen, L., Simu, S., Vainish, R., "Rich Content Control Protocol: A session control protocol instantiation", Digital Fountain document, a work in progress.

[9] Luby, M., Vicisano, L., Haken, A., "RMT BB Layered congestion control", draft-ietf-rmt-bb-lcc-00.txt, November 2000, a work in progress.

[10] Rizzo, L., "Effective Erasure Codes for Reliable Computer Communication Protocols", ACM SIGCOMM Computer Communication Review, Vol.27, No.2, pp.24-36, Apr 1997.

[11] Perrig, A., Canetti, R., Briscoe, B., Tygar, D., Song, D., "TESLA: Multicast Source Authentication Transform", draft-irtf-smug-tesla-00.txt, November, 2000, a work in progress.

[12] Rizzo, L, and Vicisano, L., "Reliable Multicast Data Distribution protocol based on software FEC techniques", Proceedings of the Fourth IEEES Workshop on the Architecture and

Implementation of High Performance Communication Systems, HPCS'97, Chalkidiki, Greece, June 1997.

[13] Speakman, T., Farinacci, D., Crowcroft, J., Gemmell, J., Lin, S., Tweedly, A., Leshchiner, D., Luby, M., Bhaskar, N., Edmonstone, R., Johnson, K., Montgomery, T., Rizzo, L., Sumanasekera, R., Vicisano, L., "PGM Reliable Transport Protocol", draft-speakman-pgm-spec-06.txt, a work in progress.

[14] Vicisano, L., Rizzo, L., Crowcroft, J., "TCP-like Congestion Control for Layered Multicast Data Transfer", IEEE Infocom '98, San Francisco, CA, Mar.28-Apr.1 1998.

[15] Vicisano, L., "Notes On a Cumulative Layered Organization of Data Packets Across Multiple groups with Different Rates", University College London Computer Science Research Note RN/98/25, Work in Progress (May 1998).

9. Authors' Addresses

Michael Luby
luby@digitalfountain.com
Digital Fountain
39141 Civic Center Drive
Suite 300
Fremont, CA, USA, 94538

Jim Gemmell
jgemmell@microsoft.com
Microsoft Research
301 Howard St., #830
San Francisco, CA, USA, 94105

Lorenzo Vicisano
lorenzo@cisco.com
cisco Systems, Inc.
170 West Tasman Dr.,
San Jose, CA, USA, 95134

Luigi Rizzo
luigi@iet.unipi.it
ACIRI/ICSI,
1947 Center St, Berkeley, CA, USA, 94704
and
Dip. Ing. dell'Informazione,
Univ. di Pisa
via Diotisalvi 2, 56126 Pisa, Italy

Mark Handley
mjh@aciri.org
ACIRI,
1947 Center St,

Berkeley, CA, USA, 94704

Jon Crowcroft
J.Crowcroft@cs.ucl.ac.uk
Department of Computer Science
University College London
Gower Street,
London WC1E 6BT, UK

10. Full Copyright Statement