



Class V Reliable Multicast

Nagatsugu Yamanouchi
Dept. Info. Science, Toho University
Chiba, Japan
yamanouc@HyperResearch.com



Class V: Reliable Multicast

1. Why not TCP?
 - Want TCP reliability, but TCP does not work
 - Problems of TCP
2. What techniques have been proposed?
 - Data Transfer Reliability
 - ARQ vs FEC FEC+ARQ
 - Scalability
 - Connection --- checking completion
 - Flow and congestion control



1. Concepts of Reliable Multicast

- Needs for TCP reliability
 - File transfer applications
 - Streaming
- Problems of TCP
 - Ack implosion
 - Connection management
 - Flow control
 - (history)



1.1-1 Needs for TCP Reliability

- Many multicast applications
 - Data and program distribution, e.g.,
 - new price list to all branches
 - software upgrade in a company
 - WEB cache data (prefetching) sharing
 - shared white board (in video conferencing)
 - Stream data delivery to many recipients, e.g.,
 - TV-like video broadcasting
 - MBONE ~ broadcasting lectures, conferences, etc.
 - video conferencing



1.1-1 Needs for TCP Reliability

- Applications want to rely on TCP reliability
 - Data reliability
 - Loss recovery by retransmission
 - Ordering and duplicate handling
 - Connection management
 - Connection establishment and termination
 - Flow and congestion control
 - Avoiding receiver's buffer overflow
 - Avoiding congestion in the middle of the route

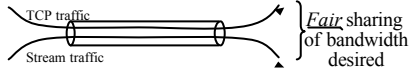


1.1-2 Reliability for file transfer

- Data reliability mandatory
 - No bit loss allowed
- Connection management mandatory
 - Session control relies on connection
- Flow/congestion control mandatory
 - for reliable network operation

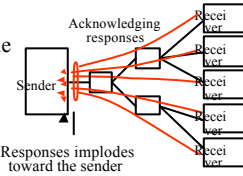
1.1-3 Reliability for streaming

- Data reliability desired even in streaming
 - “data loss allowed in multimedia stream” is not true
 - Loss seriously damages highly compressed code
- Congestion control desired
 - Co-exist with TCP ~ *TCP-friendly* control desired



1.2-1 The Problem in TCP -- Ack

- Ack implosion
 - Ack packets produces a heavy processing load at the sender
 - Ack packets becomes a heavy traffic on the links near the sender



- Limits receiver *scalability*
 - 100 receivers? 1000? 10000?
 - Any response causes implosion

1.2-1 Ack controls TCP functions

- Retransmission
 - If no ack received before timeout, retransmit the packet
- Connection management
 - Handshake during initiation and termination
 - Connection keep-alive
- Flow control
 - Receiver buffer capacity reported via Ack
- Congestion control
 - Packet loss detection reduces transmission rate

1.2-2 Common RM desired

- Now:
 - TCP is not applicable
 - No common RM that replaces TCP
 - Mostly, each application handles the problems
 - Loss recovery by application
 - Packet numbering, time out, retransmission
 - No loss recovery case
 - Accept losses in audio/video transmission
 - No flow/congestion control

1.2-2 Common RM desired

- Standardization activity toward RM transport
 - Past
 - Many efforts for a common RM failed in the past
 - Many proposed mechanisms in IETF and academia
 - e.g., J. C. Lin, S. Paul, "RMTP: A Reliable Multicast Transport Protocol", ACM SIGCOMM August 1996
 - IETF RMT -WG ~ current
 - Focus on a *collection of mechanisms* “building blocks”
 - No single mechanism satisfies the diverging requirements/situations
 - <http://www.ietf.org/html.charters/rmt-charter.html>

1.2-2 Common RM desired

- Building block approach
 - Fine, but how do you choose a block?
 - Any selection criteria?
 - How do we build up combined blocks?
 - Many unresolved problems...
- Scalability is always the key issue of IETF-RM
 - Fine, but a simpler solution is desired for a limited distribution, say up to 1000

Digression: Scalability

- Internet community emphasizes *Scalability*
- Scalability in multicast includes such as
 - Number of receivers in a group
 - mechanisms with upper limit are strongly opposed
 - Number of groups?
 - Number of senders?
 - (Size of data)

2. Reliable Multicast

- Data Transfer Reliability
 - ARQ vs FEC
 - FEC+ARQ
- Connection --- checking completion
- Flow and congestion control

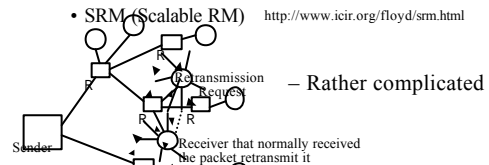
2.1-1 Data Transfer Reliability

- Retransmission-based (ARQ-based) methods
 - Simple retransmission
 - Minimize ACK size ~ “selective Nack”
 - Randomize ACK timing,
 - and so on, ...
 - e.g., draft-ietf-rmt-bb-norm-04.txt
 - Scales up to a certain size, but not to infinite
 - Starburst ~ order of 1000 receivers
 - NTT-IBM ~ around 10000 receivers

2.1-2 Data Transfer Reliability

1. Retransmission-based (ARQ-based) methods

1.1 Retransmission within a local receiver group

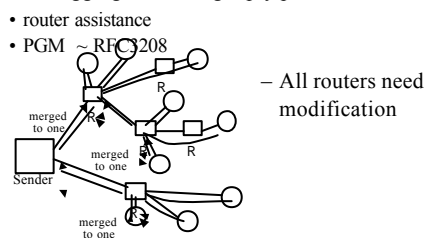


Floyd, S., et al: A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing, IEEE/ACM Trans. on Networking, Dec. 1997, Vol.5, No.6, pp.784-803.

2.1-2 Data Transfer Reliability

1. Retransmission-based (ARQ-based) methods

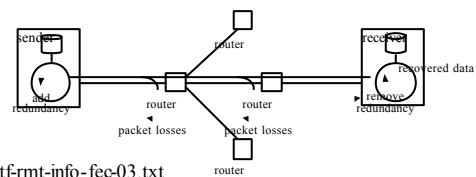
1.2 Ack aggregation along reply paths



2.1-3 Data Transfer Reliability

2. FEC-based method (Forward Error Correction)

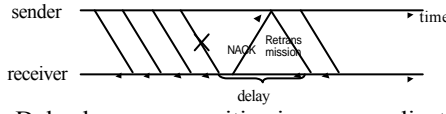
- Use *error correcting code* to recover lost packets
- (Add redundant data before transmission, recover lost part at the receiver.)



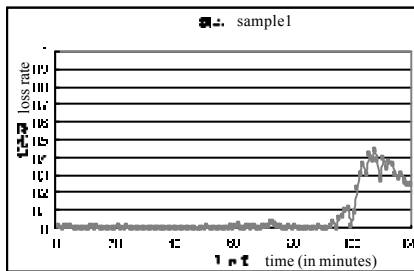
2.1-4 FEC merits vs demerits

- Merits: FEC does not need feedback
 - No Ack implosion
 - No delay for retransmission
- Problem: FEC adds constant redundancy
 - Excessive redundancy for small loss case
 - Fails if loss rate exceeds recovery limit
 - recovery limit = degree of redundancy
 - FEC cannot follow dynamic change of loss rate, which is common on the Internet

2.1-4 Retransmission delay

- Retransmission implies a delay
 
- Delay becomes sensitive in some applications
 - Realtime streaming, IP telephony
- FEC avoids this retransmission delay

2.1-4 Loss rate change example

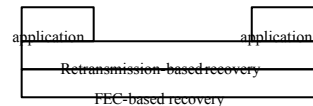


By T. Tazawa and N. Yamanouchi

2.1-5 Data Transfer Reliability

3. FEC + Retransmission

- Retransmit if FEC is not recoverable
- FEC provides a low loss rate network
 - Reduce number of Nacks ~ less implosion
- Applicable if retransmission time allows



2.2-1 Connection Management

- Some applications require connection management
 - Recognize session start by conn. establishment
 - Recognize session end by conn. termination
 - Recognize successful transfer by conn. Termination
- E.g., http, smtp, telnet, ftp, pop, ...

2.3-1 Flow/Congestion Control

- Flow control to avoid rcvr buffer overflow
- Congestion control to avoid link overflow
 - Useful to avoid unnecessary packet losses
- Difficult in multicast
 - Based on ack replies ~ ack implosion problem
 - Moverover, which rcvr/link to target?
- What to do in streaming?

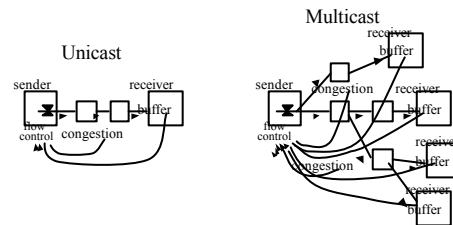
draft-ietf-rmt-bb-tfmc-01.txt
draft-ietf-rmt-bb-webrc-03.txt

2.3-2 Ack implosion problem

- Same problem as data reliability
- Flow/congestion control mandates feedback
 - Notify buffer status (progress in receiving appl) to Sender by any means
 - Reduce feedback amount as much as possible
 - Less time resolution ~ reply less frequently
 - Less spacial resolution ~ only limited sample receivers transmits a reply

2.3-3 Which receiver/link to target?

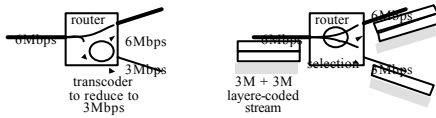
- Multicast multiple receivers and paths



- Different requests to a single throttle

2.3-4 Stream case

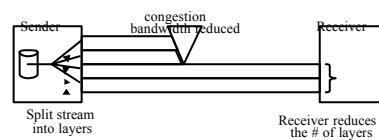
- Transmission rate fixed by video/audio rate
 - 300Kbps video needs to transmit at 300Kbps
- Changing video rate
 - On-the-fly transcoding vs Layered coding



- M-cast needs transcoder at every router infeasible

2.3-4 Layered multicast

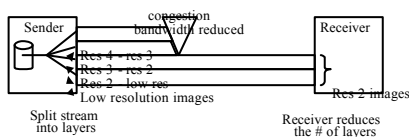
- Split stream into layers ~ one group for each layer
- Receivers selects # of layers to receive (*Join*)
 - loss detected reduce # of layers



Steven McCanne, Van Jacobson, Martin Vetterli: "Receiver-driven Layered Multicast". ACM SIGCOMM Vol.26, No.4, pp.117-130. 1996.

2.3-4 Layered multicast

- In case of video/audio, use hierarchical encode
 - Base layer: low resolution image
 - Transmitted with highest priority
 - Upper layer: (high res - low res) difference signal
 - If not delivered, low res image shows up



Summary

- Reliable multicast transport is highly desired, but not yet available
- Mechanisms have been proposed, no single mechanism suffices for wide range of appl.
 - Scalability is always the key
- Retransmission and FEC are two categories
- How to combine various methods is still an open problem