Multicast/Unicast Port Mapping Proposal

IETF 75 – July 2009

Ali Begen, Bill Ver Steeg

Introduction

- When an RTP application mixes an SSM session with unicast session, issues with port selection may arise
  - In multicast, ports are defined declaratively
  - In unicast, receivers want to choose their own ports
  - Muxing does not help as there are multiple RTP sessions involved
- E.g., in SSM distribution:
  - RTP Receiver – NACK/RAMS-R → Feedback Target (Primary RTP session)
  - Ret. Server – Ret. Packets → RTP Receiver (Unicast RTP session)
- Port selection/mapping is also an issue when a NAT device exists between the receiver and sender, even for simple RTP/RTCP repair
  - The RTCP request goes to a different port number than the RTP repair packet, so the most general NAT configurations are not self-configuring

Requirements for Solution

- General Requirements
  - Atomic, idempotent, client-driven transactions are desirable
    - Limit complexity arising from correlation of messages that do not fate-share
    - Limit amount of state maintained by the server
  - Do not introduce vectors for attacks
    - No explicit signaling of ports/addresses (w/o reverse connectivity check)
    - Do not open up a reflection attack by allowing a client to assert any IP address
  - NAT-tolerant (No ALG required)
    - Do not have transport addresses carried explicitly at the app layer
  - Do not expose IPv4/IPv6 dependencies to client-server signaling
- RAMS-Specific Requirements
  - 2xRTT stream setup (on critical path) is undesirable for RAMS

Straw-Man Proposal

- Client ascertains server port numbers, typically via SDP
- Client determines client port numbers
- Client sends message to the server port(s)

  Via a new RTCP message

  - This may require a transaction to flow from the client to the server's RTP unicast port, which implies *a subset* of RTCP-RTP port muxing *on this port*
  - If we require port muxing on this port, we should consider generally requiring port muxing on the server to reduce the number of UDP ports

    → More types of port muxing should be considered in some detail

  Perhaps using a STUN-like message, possibly combined with the previous step

Straw-Man Proposal

- Server derives address/port info from the received message via recvfrom()
- Server generates an opaque cookie that conveys the addressing information using a reversible transform
- Server sends the cookie back to the client using a new RTCP FB message
- Client includes the cookie in subsequent messages using this 5-tuple
  - Each distinct 5-tuple would have its own cookie
- Normal flows ensue, with the server using the addressing encapsulated in the opaque cookie

Summary

- We understand the multicast/unicast port mapping problem
- The problem is real
- There are a few design alternatives
- WG should study the problem in detail and propose a solution