Using Partial Offers and Partial Answers in a Multimedia Session

Adam Roach

Thursday, March 6, 2014

draft-roach-mmusic-pof-pan-02
draft-roach-mmusic-groupid-00

Changes since pof-pan-01

- Created generic group identifier mechanism in draft-roach-mmusic-groupid
    - Defined how to use that mechanism to manipulate group membership in POFs and PANs.
    - We now normatively depend on this mechanism.
- Clarified requirement for reliable, in-order delivery of POFs and PANs.
- Now allowed to add multiple streams in a single POF.

Changes since pof-pan-01 (cont)

- Made length of random “a=mid” more sensible (120 bits).
- To prevent having to “re-index” m-line indices, changes are now “quarrantined” until no POF/PAN exchange is ongoing.
- Miscellaneous editorial clarifications and minor bugfixes

One Open Issue in Draft

- Sending a POF, and then sending another POF befor getting the PAN for the first POF: allowed?
- Severely complicates procedure for computing final session state.
- I haven't heard anyone bring forth a use case that requires this ability.
- **Proposal: disallow, state that a sender must queue subsequent desired POFs until the previous one has been answered.**

Proposed Change 1: State Model

- More explicit state model: Section 5.5 will be expanded to clarify exactly what session state is used for each step of the POF/PAN exchange (cf. Kyzivat's mail to MMUSIC, 16-Feb-2014)

Proposed Change 2: Separate BUNDLE behavior

- Split BUNDLE-related statements into pof-pan-bundle draft
- This will probably be a very small document, as BUNDLE is really mentioned in only three places.
- **Question**: When we talk about splitting this, do we really mean “BUNDLE,” or do we actually mean “group”?

Proposed Change 3: Fix some terminology sloppiness

- This is a nit fix, not worth discussing; I'm just putting it up to let you know I didn't miss it.
- Editorial clarification: formally refer to "o= line <session-version>" instead of "version" everywhere.

Group ID Mechanism, overview

- Adds new “group-id” attribute to *session level*, to identify the group defined immediately after.
- Adds new “in-group” attribute to *media level*, to identify group(s) the m-line is in.

```
v=0
o=Laura 289083124 289083124 IN …
c=IN IP4 192.0.2.1
t=0 0
a=group-id:abc
a=group:LS 1 2
m=audio 30000 RTP/AVP 0
a=mid:1
a=in-group:LS abc
m=audio 30000 RTP/AVP 8
a=mid:2
a=in-group:LS abc
```

Open Issue: SDP Attribute Ordering

- Mechanism relies on “group-id” attribute ordering to be significant relative to “group” attributes.
- It is hypothesized that some legacy SDP parsers may re-order attributes
  - Although it’s not clear that this is valid behavior, as RFC 4566 (and RFC 2327 before it) assigned significance to attribute ordering.

SDP Attribute Ordering, cont

- **Option 1**: Leave mechanism as-is, adding protections that attempt to detect mismatches and abandon the naming (and anything that relies on it, like POF/PAN)
- **Option 2**: Throw RFC 5888 groups under the bus, forever abandoning hope of backwards compatibility
- **Option 3**: Rework syntax to define group ID as “phantom” MID in “group” attribute

Proposal: Option 1

- The hypothesized misbehaving nodes aren't even known to exist; it seems premature to try to optimize around them.
- Nodes implementing this mechanism will necessarily be new, and it's not unreasonable to require them to have well-behaved SDP parsers.
- With combination of group-id and in-group attributes, detecting mismatches due to misbehaving intermediaries is trivial.

Next Steps

- Feedback has been good, but more sparse than I'd like.
- Before I invest too much more time in this: does this feel like an approach that's likely to lead to success?