Trickle ICE

Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol

draft-rescorla-mmusic-ice-trickle

Eric Rescorla
Justin Uberti
Emil Ivov
Why

Simply stated: makes call setup faster

- Don't need to wait for all candidates/servers to respond
- Encourages use of multiple STUN/TURN servers
- Savings occur on both sides
- Removes need for magic timeouts

In a typical app, call setup is improved by 500+ ms in:

- 25% of calls (globally)
- 50% of calls (developing markets)
- Even more if service is not multi-region
The Problem

Vanilla ICE Operation as per RFC 5245
The Solution

send candidates as you get them

.offer and ICE description (and host candidates)

answer and ICE description (and host candidates)

...more candidates and connectivity checks...

Trickle ICE
Current state

- Trickle ICE is already implemented in existing signaling protocols and real-world apps
  - XMPP (XEP-0176), JSEP (Chrome)
  - Google Talk, Empathy, etc

- But we need to nail down the exact interactions with RFC 5245
  - draft-rescorla-mmusic-ice-trickle does this
Relationship to SIP, Offer/Answer and SDP

- Extends existing ICE Offer/Answer Model
  - Defines semantics for Trickle ICE
  - ICE credentials are exchanged via offer/answer
  - More ICE candidates can be sent after credentials

- Defines SDP for negotiating Trickle ICE
  - well, not yet but in v01

- Outer signaling is left abstract
  - No SIP message for candidates defined by this spec
Details for Trickle ICE
(what's currently in the spec)

- How to act upon learning additional candidates

- How to indicate that all candidates have been gathered

- How to handle ICE check list states when trickling

- How to keep checks synchronized from both ends (since we can no longer rely on stream and pair ordering)
Why end-of-candidates is needed

Alice

Delivers All Candidates

Delivers Some Candidates

Connectivity Checks Fail

Delivers Additional Candidates

New Connectivity Checks Succeed

end-of-candidates

Bob
Backwards Compatibility

- A non-Trickle callee can't handle a Trickle offer

- Full Trickle
  - If you know remote side supports Trickle ICE (via XMPP Disco, Cap Neg, or other out-of-band mechanism)
  - Both caller and callee can trickle

- Half Trickle
  - Caller cannot trickle, but answerer can
  - Still get half the latency savings
SDP Details
(Work in Progress)

- Need SDP to indicate Trickle support
  - a=ice-options:trickle

- Need way to generate valid SDP with no candidates
  - For cases where we don’t want to send host candidates for privacy reasons
    - c=0.0.0.0
    - m=1
    - a=candidate:0.0.0.0 (← do we need this?)