TRICKLE ICE

TRICKLE ICE

draft-ietf-mmusic-trickle-ice

Emil Ivov, Eric Rescorla, Justin Uberti, Peter Saint-Andre

99%

Special thanks to: Taylor Brandstetter

∆s since 02

Trickle ICE plus ICE restarts #3

Also, candidate trickling needs to be correlated to a specific ICE negotiation session, so that if there is an ICE restart, any delayed updates for a previous session can be recognized as such and ignored by the receiving party. For example, applications that choose to signal candidates via SDP may include a ufrag value in the SDP that represents candidates such as:

```
  a=candidate:1 1 UDP 2130706431 2001:db8::1 5000 typ host ufrag 8hhY
```

Or as another exmaple, WebRTC implementations may include a ufrag in the JavaScript objects that represent candidates.

Unfreezing procedures (update to decision at IETF 95) #2

A Trickle ICE agent initially considers all check lists to be frozen. It then inspects the first check list and attempts to unfreeze all candidates it has received so far that belong to the first component on the first media stream (i.e., the first media stream that was reported to the ICE implementation from the using application). **If that first component of the first media stream does not contain candidates for one or more of the currently known pair foundations, and if candidate pairs already exist for that foundation in one of the following components or media streams, then the agent unfreezes the first of those.**

Update candidate deduplication based on decision at IETF 95: choose highest priority #6

Update candidate deduplication based on decision at IETF 95: choose highest priority #6

Taylor Brandstetter: this means that peer-reflexive would always win. problem!

Decision: highest prio wins, except for peer-reflexive which always lose the race

Update candidate deduplication based on decision at IETF 95: choose highest priority #6

Taylor Brandstetter: this means that peer-reflexive would always win. problem!

Decision: highest prio wins, except for peer-reflexive which always lose the race

```
Once this is done, the agent examines the check list looking for another pair
that would be redundant with the new one. If such a pair exists and its type
is not peer reflexive, the pair with the higher priority is kept and the one
with the lower priority is discarded. If, on the other hand, the type of the
pre-existing pair is peer reflexive, the agent MUST replace it with the new
candidate it received, regardless of their priorities.

    Note: Replacing pre-existing pairs with seemingly equivalent higher-
    priority ones helps guarantee that both agents will have the same view of
    candidate priorities. This is particularly important during aggressive
    nomination, when priority is sometimes the only way a controlled agent
    can determine the selected pair. It is for that same reason that peer-
    reflexive candidates need to always be updated if equivalent alternatives
    are received through signalling.
```

anything else we can do for you?