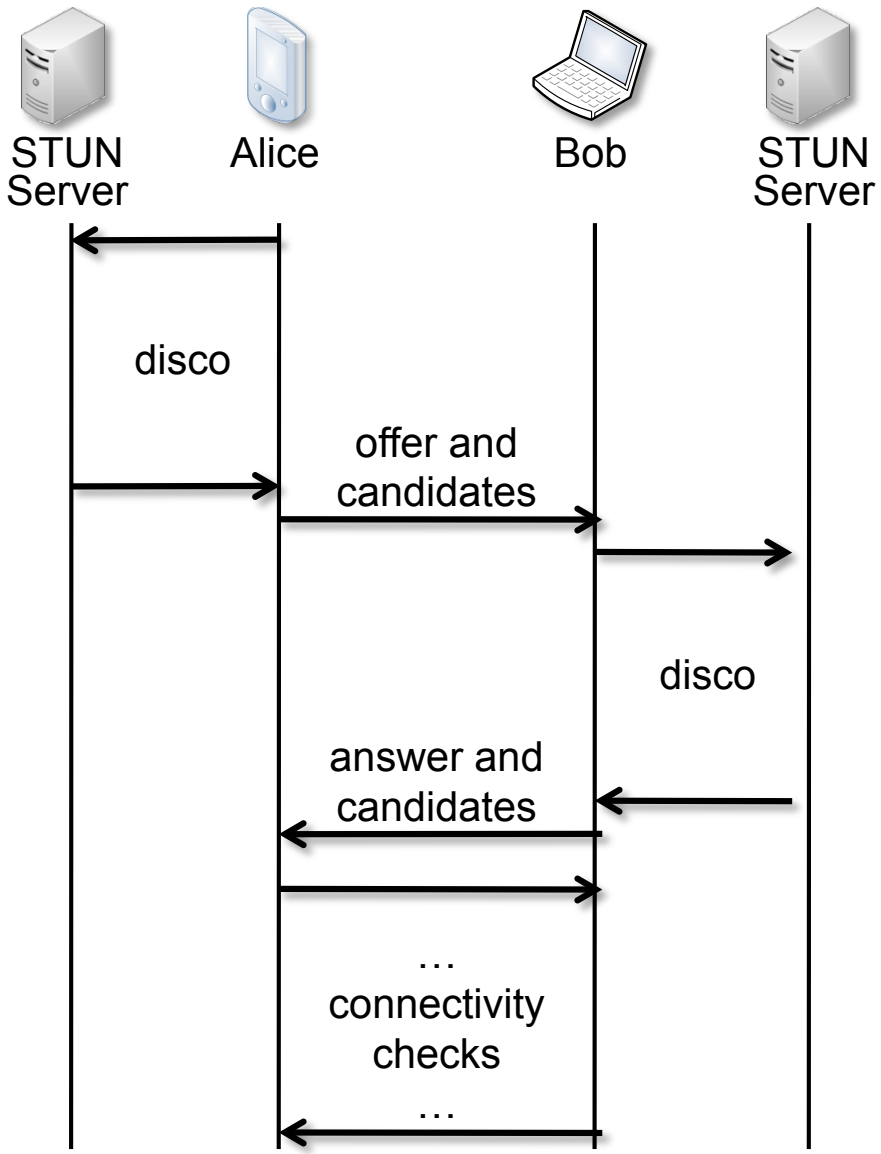
Trickle ICE

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

draft-ivov-mmusic-trickle-ice

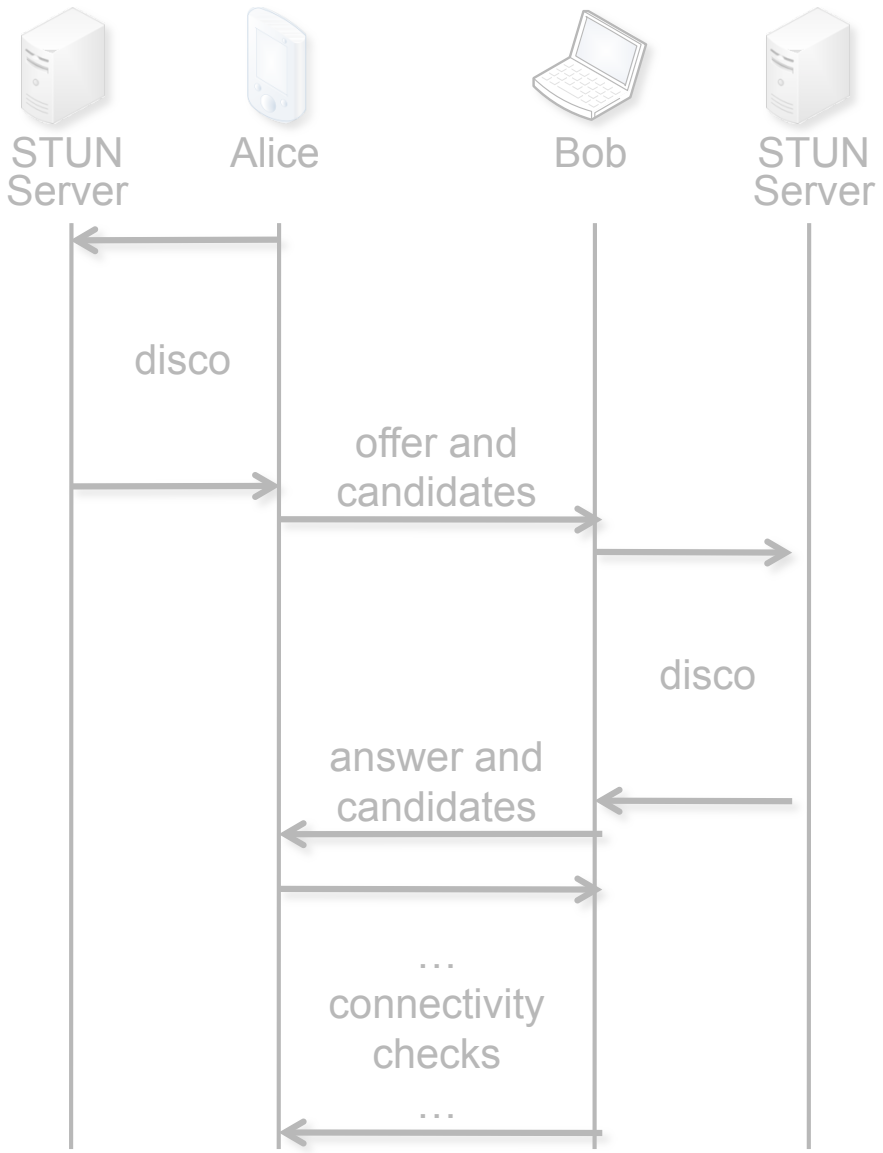
Eric Rescorla
Justin Uberti
Emil Ivov

Reminder: Vanilla ICE

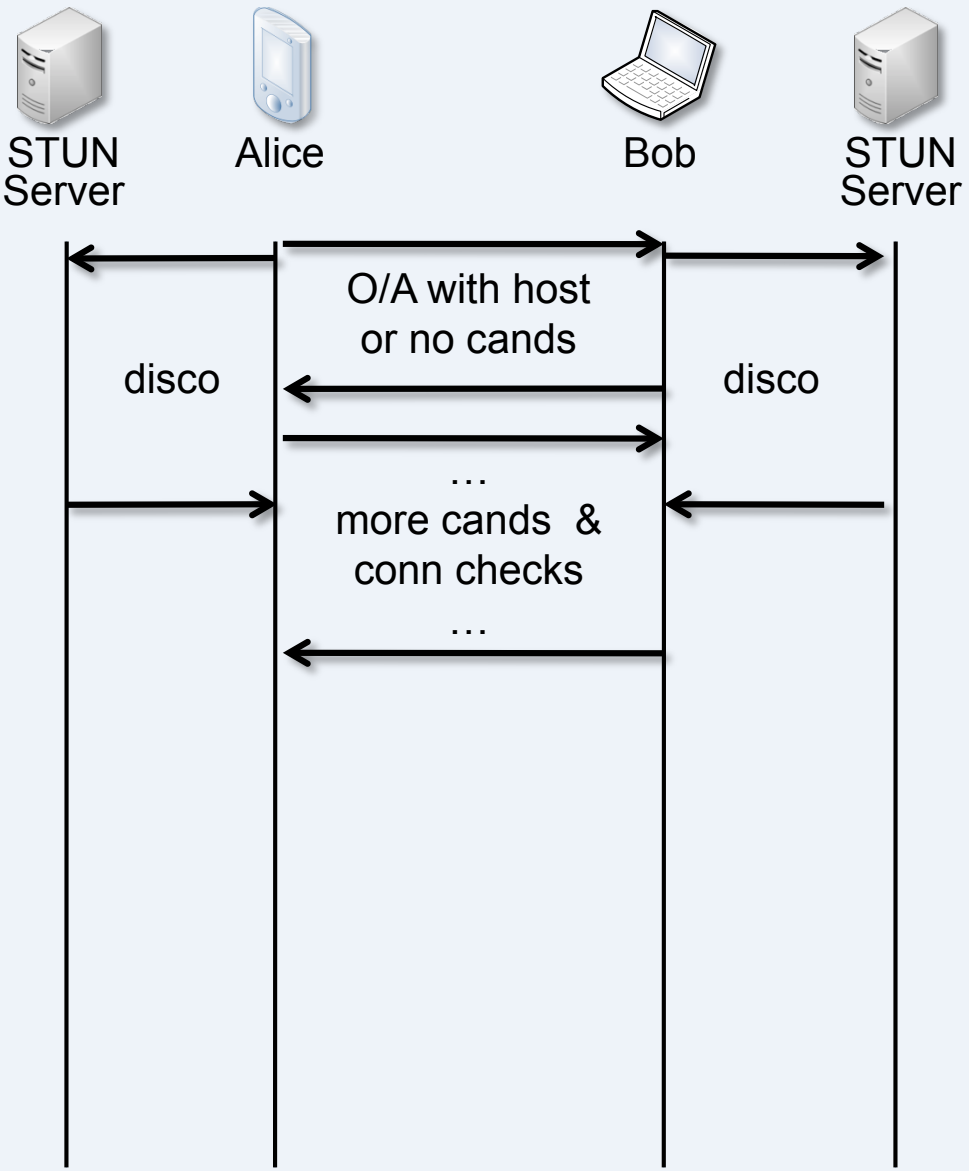


Vanilla ICE as per RFC 5245

Reminder: Vanilla ICE vs Trickle ICE



Vanilla ICE as per RFC 5245



Trickle ICE

Decisions from the Boston Interim (1/3)

- Advertising support for trickle ICE:

  a=ice-options:trickle

  maybe also add an a=ice-options:trickle-on … or maybe not
  (list comment from Ari)

- Offers and answers with no candidates:

```
c=IN IP4 0.0.0.0
m=audio 1 RTP/AVP 0 96
```

- Change c= line for O/As with no cands:

```
- c=IN IP4 0.0.0.0
- m=audio 1 RTP/AVP 0 96
+ c=IN IP6 ::
+ m=audio 9 RTP/AVP 0 96
```

Decisions from the Boston Interim (2/3)

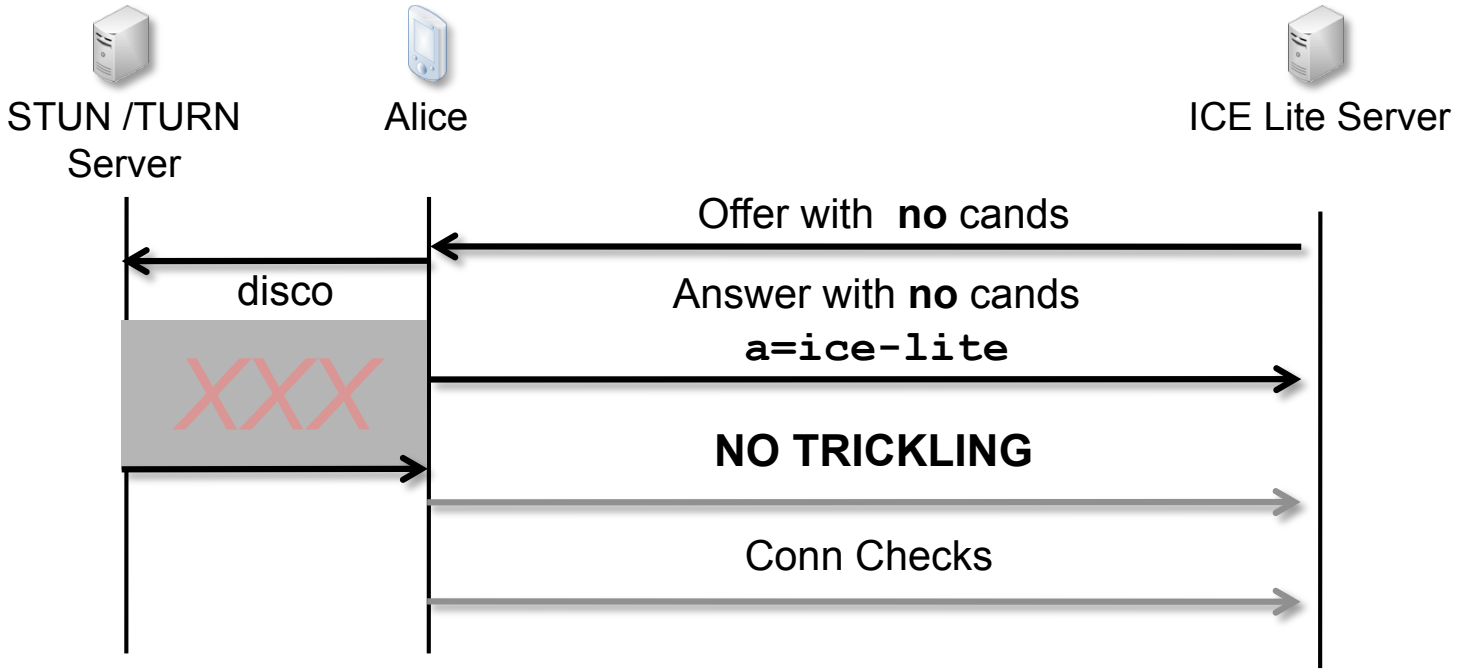
- New candidates and end-of-candidates

```
…
a=candidate:1 1 UDP 1234 1.2.1.4 5000 typ host
a=candidate:2 1 UDP 5678 6.1.2.3 5000 typ srflx
a=end-of-candidates
```

- Remove use of m= line index when sending trickled candidates. Will use MID only
- Always send end-of-candidates unless ICE processing has ended
- MUST NOT send candidates after that and MUST do an ICE restart to change
- Add a reference to the SIP usage document
- Specify "end-of-candidates” as media level (obviously can be session too)

Decisions from the Boston Interim (3/3)

- Add wording and an example explaining that ICE lite agents don't need to see the trickling



- Open Issue: Does it make sense to point that disco here is only necessary for relayed cands?

Starting Checks and Unfreezing Pairs Vanilla ICE Reminder

		IPv4 host	IPv6 host	srflx	relayed
	Str.Cmp	Foundation1	Foundation2	Foundation3	Foundation4
CheckList.1	Audio.1	192.168.0.1:5000	[2001:660::1]:5000	130.129.0.1:5000	8.9.0.1:5000
	Audio.2	192.168.0.1:5001	[2001:660::1]:5001	130.129.0.1:5001	8.9.0.1:5001
CheckList.2	Video.1	192.168.0.1:5002	[2001:660::1]:5002	130.129.0.1:5002	8.9.0.1:5002
	Video.2	192.168.0.1:5003	[2001:660::1]:5003	130.129.0.1:5003	8.9.0.1:5003

For simplicity, imagine that:

- all local candidates are paired with a single remote one: 192.168.0.2:5000
- IPv6 is backward compatible

Starting Checks and Unfreezing Pairs Vanilla ICE Reminder

		IPv4 host	IPv6 host	srflx	relayed
	Str.Cmp	Foundation1	Foundation2	Foundation3	Foundation4
CheckList.1	Audio.1	**192.168.0.1:5000**	**[2001:660::1]:5000**	**130.129.0.1:5000**	**8.9.0.1:5000**
	Audio.2	192.168.0.1:5001	[2001:660::1]:5001	130.129.0.1:5001	8.9.0.1:5001
CheckList.2	Video.1	192.168.0.1:5002	[2001:660::1]:5002	130.129.0.1:5002	8.9.0.1:5002
	Video.2	192.168.0.1:5003	[2001:660::1]:5003	130.129.0.1:5003	8.9.0.1:5003

[RFC5245] says that by default everything is Frozen and then:

```
The agent examines the check list for the first media stream.
For that media stream:

      *  For all pairs with the same foundation, it sets the
state of the pair with the lowest component ID to Waiting.
```

Starting Checks and Unfreezing Pairs Vanilla ICE Reminder

		IPv4 host	IPv6 host	srflx	relayed
	Str.Cmp	Foundation	Foundation2	Foundation3	Foundation4
CheckList.1	Audio.1	**192.168.0.1:5000**	**[2001:660::1]:5000**	**130.129.0.1:5000**	**8.9.0.1:5000**
	Audio.2	**192.168.0.1:5001**	[2001:660::1]:5001	130.129.0.1:5001	8.9.0.1:5001
CheckList.2	Video.1	192.168.0.1:5002	[2001:660::1]:5002	130.129.0.1:5002	8.9.0.1:5002
	Video.2	192.168.0.1:5003	[2001:660::1]:5003	130.129.0.1:5003	8.9.0.1:5003

1. The agent changes the states for all other Frozen pairs for the same media stream and same foundation to Waiting.

Starting Checks and Unfreezing Pairs Vanilla ICE Reminder

		IPv4 host	IPv6 host	srflx	relayed
	Str.Cmp	Foundation	Foundation2	Foundation3	Foundation4
CheckList.1	Audio.1	**192.168.0.1:5000**	**[2001:660::1]:5000**	**130.129.0.1:5000**	**8.9.0.1:5000**
	Audio.2	**192.168.0.1:5001**	[2001:660::1]:5001	130.129.0.1:5001	8.9.0.1:5001
CheckList.2	Video.1	**192.168.0.1:5002**	[2001:660::1]:5002	130.129.0.1:5002	8.9.0.1:5002
	Video.2	**192.168.0.1:5003**	[2001:660::1]:5003	130.129.0.1:5003	8.9.0.1:5003

2. If there is a pair in the valid list for every component of this media stream. The agent examines the check list for each other media stream in turn :

   * the state of all pairs in the check list whose foundation matches a pair in the valid list under consideration is set to Waiting

Starting Checks and Unfreezing Pairs Trickle ICE (Open Issue)

		IPv4 host	IPv6 host	srflx	relayed
	Str.Cmp	Foundation1	Foundation2	Foundation3	Foundation4
CheckList.1	Audio.1		[2001:660::1]:5000		
	Audio.2			130.129.0.1:5001	8.9.0.1:5001
CheckList.2	Video.1	192.168.0.1:5002			
	Video.2		[2001:660::1]:5003		8.9.0.1:5003

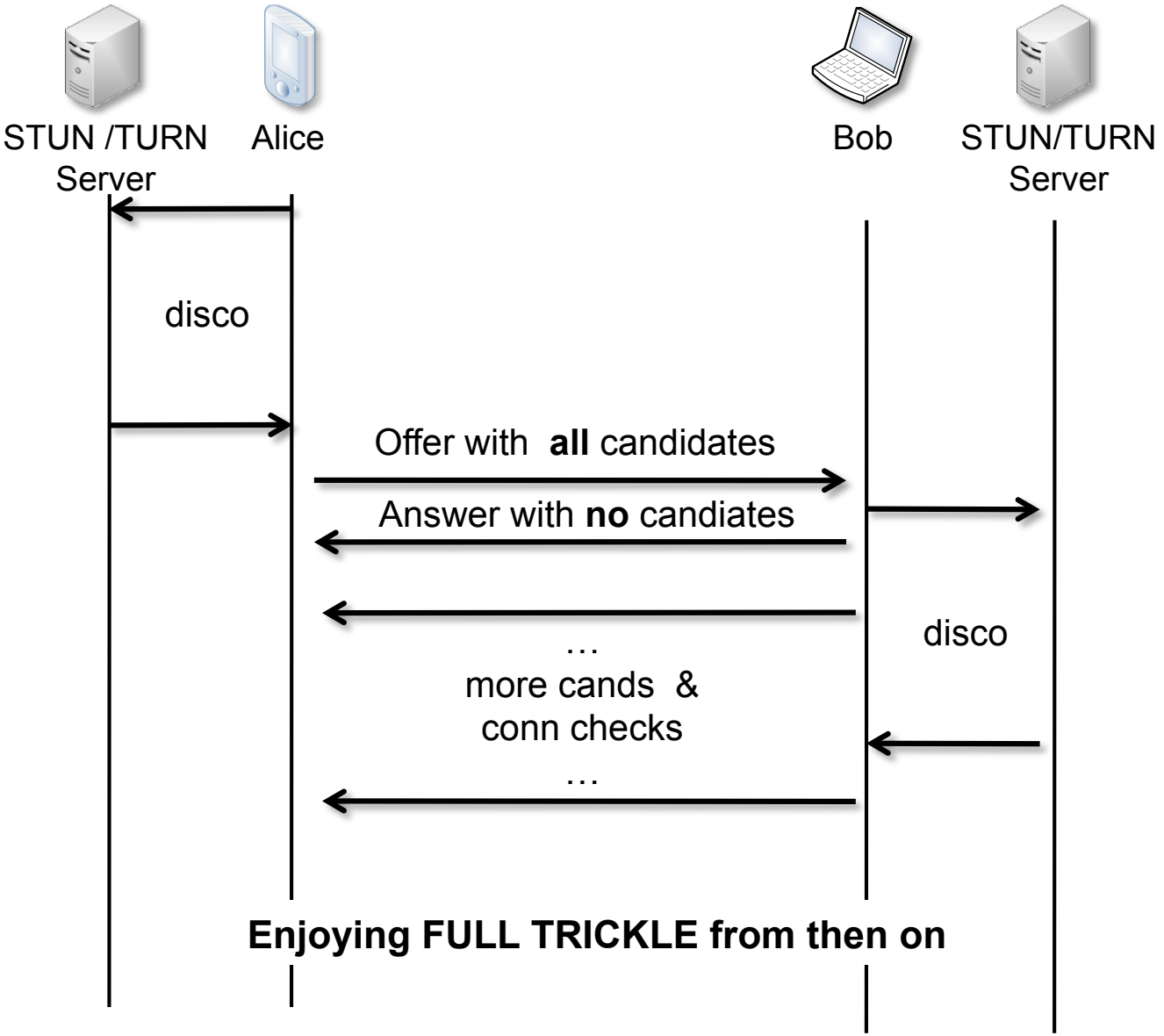
- With trickle ICE we start with the first non empty list but then…
- pairs will not necessarily be crated on a list by list basis.
- Therefore, can we just concentrate on foundations?

Reminder: Ending Checks

- Vanilla ICE: Every time a conn check completes thou shalt update states and fail a check list if:
  - all of its pairs are either in the Failed or Succeeded state;
  - at least one of the components of the media stream has no pairs in its valid list.

- Trickle ICE adds the following conditions:
  - all candidate harvesters have completed and the agent is not expecting to learn any new candidates;
  - the remote agent has sent an end-of-candidates indication for that check list

Half Trickle



Open Issues
MID vs Stream Index

- Do we really need the stream index option?

- Possible application syntax (do we want to spec this)?

For example:

```
a=mid:1
a=candidate:1 1 UDP 16582 12.18.10.3 5000 typ host
a=candidate:2 1 UDP 16584 96.1.2.3 5000 typ srflx
a=end-of-candidates
a=mid:2
a=candidate:2 1 UDP 16915 96.1.2.3 5002 typ srflx
```

Open Issues

Session or media level end-of-candidates

```
c=IN IP4 12.18.10.3
a=end-of-candidates
m=audio 5000 RTP/AVP 0 96
a=candidate:1 1 UDP 16582 12.18.10.3 5000 typ host
m=video 5000 RTP/AVP 0 96
a=candidate:2 1 UDP 16915 96.1.2.3 5002 typ srflx
```

VS

(our preference)

```
c=IN IP4 12.18.10.3
m=audio 5000 RTP/AVP 0 96
a=candidate:1 1 UDP 16582 12.18.10.3 5000 typ host
a=end-of-candidates
m=video 5000 RTP/AVP 0 96
a=candidate:2 1 UDP 16915 96.1.2.3 5002 typ srflx
a=end-of-candidates
```

Appendix:

A SIP Usage for Trickle ICE (1/3)

- SIP Applications would always do half trickle unless explicitly configured otherwise

- Trickling will happen with in-dialog SIP INFO requests as per RFC 6086.

- The INFO Package token name for this package is "trickle-ice”
  - Does not mandate GRUU support

- Does not remove the requirement for doing a re-INVITE upon completion of ICE processing.

Appendix:
A SIP Usage for Trickle ICE (2/3)

```
INFO sip:alice@example.com SIP/2.0
...
Info-Package: trickle-ice
Content-type: application/sdpfrag
Content-Disposition: Info-Package
Content-length: ...

a=mid:1
a=candidate:1 1 UDP 1658497328 192.168.100.33 5000 typ host
a=candidate:2 1 UDP 1658497328 96.1.2.3 5000 typ srflx
a=mid:2
a=candidate:2 1 UDP 1658497328 96.1.2.3 5002 typ srflx
a=end-of-candidates
```

Content type is application/sdpfrag defined in draft-ivov-dispatch-sdpfrag (WIP)

Appendix:
A SIP Usage for Trickle ICE (3/3)