XEP-0371: Jingle ICE Transport Method

Peter Saint-Andre
mailto:stpeter@stpeter.im
xmpp:stpeter@jabber.org
https://stpeter.im/

2021-03-04
Version 0.3.1

Status	Type	Short Name
Deferred	Standards Track	jingle-ice

This specification defines a Jingle transport method that results in sending media data using datagram associations via the User Datagram Protocol (UDP) or using end-to-end connections via the Transport Control Protocol (TCP). This transport method is negotiated via the Interactive Connectivity Establishment (ICE) methodology (which provides robust NAT traversal for media traffic) and also supports the ability to exchange candidates throughout the life of the session, consistent with so-called ”Trickle ICE” (draft-ietf-ice-trickle).

Legal

Copyright

Permissions

Warranty

NOTE WELL: This Specification is provided on an ”AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.

Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall the XMPP Standards Foundation or any author of this Specification be liable for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if the XMPP Standards Foundation or such author has been advised of the possibility of such damages.

Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF’s Intellectual Property Rights Policy (a copy of which can be found at <https://xmpp.org/about/xsf/ipr-policy> or obtained by writing to XMPP Standards Foundation, P.O. Box 787, Parker, CO 80134 USA).

Contents

1 Introduction

Jingle (XEP-0166) [1] defines a framework for negotiating and managing out-of-band data sessions over XMPP. In order to provide a flexible framework, the base Jingle specification defines neither data transport methods nor application formats, leaving that up to separate specifications.
The current document defines a transport method for establishing and managing data exchanges between XMPP entities by means of the Interactive Connectivity Establishment (ICE) methodology specified in RFC 8445 [2]. The Jingle usage of ICE was also the first technology to send ICE candidates incrementally, a technique that has since become known as "Trickle ICE" Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol [3].
The process for ICE negotiation is largely the same in Jingle as it is in RFC 8445 [4]. There are several differences:

- Instead of using the Session Initiation Protocol (SIP) as the signalling channel, Jingle uses XMPP as the signalling channel.
- Syntax from the Session Description Protocol (see RFC 4566 [5]) is mapped to an XML syntax suitable for sending over the XMPP signalling channel.
- In Jingle, lists of "preferred" candidates are typically sent in the Jingle session-initiate and session-accept messages, in a way that is consistent with the SDP offer / answer model described in RFC 3264 [6] and the process described in RFC 8445 [7].
- Candidates can also be sent in separate transport-info messages either before sending or receiving the session-accept message (to expedite negotiation) or after media begins to flow (to find modify existing candidates, find superior candidates, or adjust to changing network conditions). This usage, which has been part of the Jingle ICE transport method since 2005, has since come to be known as "Trickle ICE"; as defined here the usage is consistent with the IETF specification for Trickle ICE Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol [8].

As originally defined in XEP-0166 and then Jingle ICE-UDP Transport Method (XEP-0176) [9] the use of ICE in Jingle applied only to negotiations that established a User Datagram Protocol

[1] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.
[2] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.
[3] Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol <http://tools.ietf.org/html/draft-ietf-ice-trickle/>.
[4] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.
[5] RFC 4566: SDP: Session Description Protocol <http://tools.ietf.org/html/rfc4566>.
[6] RFC 3264: An Offer/Answer Model with the Session Description Protocol (SDP) <http://tools.ietf.org/html/rfc3264>.
[7] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.
[8] Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol <http://tools.ietf.org/html/draft-ietf-ice-trickle/>.
[9] XEP-0176: Jingle ICE-UDP Transport Method <https://xmpp.org/extensions/xep-0176.html>.

association (see RFC 768 [10]) and thus resulted in a Jingle datagram transport suitable for media applications where some packet loss is tolerable (e.g., audio and video). However, since the publication of RFC 6544 [11] in 2012 it has also been possible to exchange Transmission Control Protocol (see RFC 793 [12]) candidates during ICE negotiation. Therefore this document expands the use of ICE in Jingle to also establish a TCP connection and thus result in a Jingle stream transport suitable for media applications where packet loss cannot be tolerated (e.g., file transfer). To reduce the possibility of confusion, the expanded definition provided here is specified in a new XEP, which is intended to supersede XEP-0176.

2 Glossary

The reader is referred to RFC 8445 [13] and draft-ietf-ice-trickle for a description of various terms used in the context of ICE. Those terms are not reproduced here.

3 Requirements

The Jingle transport method defined herein is designed to meet the following requirements:

1. Make it possible to establish and manage out-of-band connections between two XMPP entities, even if they are behind Network Address Translators (NATs) or firewalls.
2. Enable use of UDP or TCP as the transport protocol.
3. Make it relatively easy to implement support in standard Jabber/XMPP clients.
4. Where communication with non-XMPP entities is needed, push as much complexity as possible onto server-side gateways between the XMPP network and the non-XMPP network.

4 Jingle Conformance

In accordance with Section 10 of Jingle (XEP-0166) [14], this document specifies the following information related to the Jingle ICE transport method:

[10] RFC 768: User Datagram Protocol <http://tools.ietf.org/html/rfc0768>.

[11] RFC 6544: TCP Candidates with Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc6544>.

[12] RFC 793: Transmission Control Protocol <http://tools.ietf.org/html/rfc0793>.

[13] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[14] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.

1. The transport negotiation process is defined in the Protocol Description section of this document.

2. The semantics of the <transport/> element are defined in the ICE Negotiation section of this document.

3. Depending on the kinds of candidates exchanged, successful negotiation of this method results in use of a datagram transport (suitable for applications where some packet loss is tolerable, such as audio and video) or of a streaming transport (suitable for applications where packet loss is not tolerable, such as file transfer).

4. If multiple components are to be communicated by the application type that uses the transport, the transport shall support those components and assign identifiers for them as described in the specification that defines the application type.

5 Protocol Description

5.1 Overall Flow

The overall protocol flow for negotiation of the Jingle ICE Transport Method is as follows (note: many of these events happen simultaneously, not in sequence).

```
INITIATOR                              RESPONDER
    |                                      |
    |  Jingle session-initiate stanza      |
    |  (with zero or more candidates)      |
    |------------------------------------->|
    |  Jingle ack (XMPP IQ-result)         |
    |<-------------------------------------|
    |  Jingle session-accept stanza        |
    |  (with one or more candidates)       |
    |<-------------------------------------|
    |  Jingle ack (XMPP IQ-result)         |
    |------------------------------------->|
    |  multiple STUN Binding Requests      |
    |<====================================>|
    |  multiple STUN Binding Results       |
    |<====================================>|
    |<=========MEDIA NOW FLOWS============>|
    |  optional Jingle transport-info      |
    |  stanzas (one candidate per stanza)  |
    |<------------------------------------>|
    |                                      |
```

Note: The examples in this document follow the scenario described in Section 15 of RFC 8445 [15], except that we substitute the Shakespearean characters "Romeo" and "Juliet" for the generic entities "L" and "R".

5.2 Session Initiation

In order for the initiator in a Jingle exchange to start the negotiation, it sends a Jingle "session-initiate" stanza that includes at least one content type, as described in Jingle (XEP-0166) [16]. If the initiator wishes to negotiate the ICE transport method for an application format, it MUST include a <transport/> child element qualified by the 'urn:xmpp:jingle:transports:ice:0' namespace (see Namespace Versioning regarding the possibility of incrementing the version number). This element SHOULD in turn contain one <candidate/> element for each of the initiator's higher-priority transport candidates as determined in accordance with the ICE methodology, but MAY instead be empty (with each candidate to be sent as the payload of a transport-info message).

Listing 1: Initiation

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='ixt174g9'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='session-initiate'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='this-is-the-audio-content'>
      <description xmlns='urn:xmpp:jingle:apps:rtp:1' media='audio'>
        <payload-type id='96' name='speex' clockrate='16000'/>
        <payload-type id='97' name='speex' clockrate='8000'/>
        <payload-type id='18' name='G729'/>
        <payload-type id='0' name='PCMU' />
        <payload-type id='103' name='L16' clockrate='16000' channels='
          2'/>
        <payload-type id='98' name='x-ISAC' clockrate='8000'/>
      </description>
      <transport xmlns='urn:xmpp:jingle:transports:ice:0'
                 pwd='asd88fgpdd777uzjYhagZg'
                 ufrag='8hhy'
                 ice2='true'>
        <candidate component='1'
                   foundation='2B78DADC1A9E'
                   generation='0'
                   id='el0747fg11'
```

[15] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[16] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.

```
                   ip='10.0.1.1'
                   network='1'
                   port='8998'
                   priority='2130706431'
                   protocol='udp'
                   type='host'/>
        <candidate component='1'
                   foundation='58AA96B8FA5A'
                   generation='0'
                   id='y3s2b30v3r'
                   ip='192.0.2.3'
                   network='1'
                   port='45664'
                   priority='1694498815'
                   protocol='udp'
                   rel-addr='10.0.1.1'
                   rel-port='8998'
                   type='srflx'/>
      </transport>
    </content>
  </jingle>
</iq>
```

5.3 Syntax

The <transport/> element’s ’pwd’ and ’ufrag’ attributes MUST be included whenever sending one or more candidates to the other party, e.g., in a session-initiate, session-accept, transport-info, content-add, or transport-replace message. The values for these attributes are separately generated for both the initiator and the responder, in accordance with RFC 8445 [17] and as shown in the examples.

’ice2’ attribute tells about compliancy with RFC 8445 [18]. If the attribute is not set or set to ’false’ in <transport/> element, the recipient can assume RFC 5245 [19]. The value of the attribute may not be changed during lifetime of the transport instance, but it’s not an error to skip the attribute in consequent transport-info updates.

The attributes of the <transport/> element are as follows.

Name	Description	SDP Syntax	Example
pwd	A Password as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>..	a=ice-pwd line	asd88fgpdd777uzjYhagZg

[17] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[18] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[19] RFC 5245: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc5245>.

Name	Description	SDP Syntax	Example
ufrag	A Username Fragment as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>..	a=ice-ufrag line	8hhy
ice2	ice2 option as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>..	a=ice-options:ice2	true

The attributes of the <candidate/> element are as follows.

Name	Description	SDP Syntax	Example
component	A Component ID as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>..	Component ID value in a=candidate line	1
foundation	A Foundation as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.. (Note that version 1.0 of this specification container an error, whereby the data type for the Jingle 'foundation' attribute was defined as xs:unsignedByte; in version 1.1 this was corrected to xs:string, however some existing implementations might not use or expect strings.)	Foundation value in a=candidate line	2B78DADC1A9E

Name	Description	SDP Syntax	Example
generation	An index, starting at 0, that enables the parties to keep track of updates to the candidate throughout the life of the session. For details, see the ICE Restarts section of this document.	extended name/value pair in a=candidate line	0
id	A unique identifier for the candidate.	N/A	el0747fg11
ip	The Internet Protocol (IP) address for the candidate transport mechanism; this can be either an IPv4 address or an IPv6 address.	IP Address value in a=candidate line	192.0.2.3
network	An index, starting at 0, referencing which network this candidate is on for a given peer (used for diagnostic purposes if the calling hardware has more than one Network Interface Card).	N/A	0
port	The port at the candidate IP address.	Port value in a=candidate line	45664

Name	Description	SDP Syntax	Example
priority	A Priority as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.. In accordance with the rules specified in Section 5.1.2 of RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>., the priority values shown in the examples within this document have been calculated as follows. The ”type preference” for host candidates is stipulated to be ”126”, ”110” for peer reflexive and for server reflexive candidates ”100”. The ”local preference” for network 0 is stipulated to be ”4096”, for network 1 ”2048”, and for network 2 ”1024”.	Priority value in a=candidate line	2130706431
protocol	The protocol to be used. The values allowed by this specification are ”udp” (see RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.) and ”tcp” (see RFC 6544 RFC 6544: TCP Candidates with Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc6544>.).	Transport protocol field in a=candidate line	udp

Name	Description	SDP Syntax	Example
rel-addr	A related address as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>..	Value of raddr attribute in a=candidate line	10.0.1.1
rel-port	A related port as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>..	Value of rport attribute in a=candidate line	8998
tcptype	A TCP candidate type as defined in RFC 6544 RFC 6544: TCP Candidates with Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc6544>.. The allowable values are ”active” for TCP active candidates, ”passive” for TCP passive candidates, and ”so” for TCP simultaneous-open candidates.	Value of tcptype attribute in a=candidate line	so

Name	Description	SDP Syntax	Example
type	An ICE candidate type as defined in RFC 8445 RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.. The allowable values are ”host” for host candidates, ”prflx” for peer reflexive candidates, ”relay” for relayed candidates, and ”srflx” for server reflexive candidates. Note that TCP candidate types (RFC 6544 RFC 6544: TCP Candidates with Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc6544>.) are handled via the ’tcptype’ attribute.	Value of typ attribute in a=candidate line	srflx

5.4 Response

As described in Jingle (XEP-0166) [20], to acknowledge receipt of the session initiation request, the responder immediately returns an IQ-result.

Listing 2: Responder acknowledges receipt of session-initiate request

```
<iq from='juliet@capulet.example/yn0cl4bnw0yr3vym'
    id='ixt174g9'
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='result'/>
```

Depending on the application type, a user agent controlled by a human user might need to wait for the user to affirm a desire to proceed with the session before continuing. When the user agent has received such affirmation (or if the user agent can automatically proceed for any reason, e.g., because no human intervention is expected or because a human user has configured the user agent to automatically accept sessions with a given entity), it returns a

[20] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.

Jingle session-accept message. This message MUST contain a <transport/> element qualified by the ’urn:xmpp:jingle:transports:ice:0’ namespace, which SHOULD in turn contain one <candidate/> element for each ICE candidate generated by or known to the responder, but MAY instead be empty (with each candidate to be sent as the payload of a transport-info message).
Note: See the Security Considerations section of this document regarding the exposure of IP addresses by the responder’s client.

Listing 3: Responder accepts the session request

```
<iq from='juliet@capulet.example/yn0cl4bnw0yr3vym'
    id='rw782g55'
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='session-accept'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          responder='juliet@capulet.example/yn0cl4bnw0yr3vym'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='this-is-the-audio-content'>
      <description xmlns='urn:xmpp:jingle:apps:rtp:1' media='audio'>
        <payload-type id='97' name='speex' clockrate='8000'/>
        <payload-type id='18' name='G729'/>
      </description>
      <transport xmlns='urn:xmpp:jingle:transports:ice:0'
                 pwd='YH75Fviy6338Vbrhrlp8Yh'
                 ufrag='9uB6'>
        <candidate component='1'
                   foundation='2B78DADC1A9E'
                   generation='0'
                   id='or2ii2syr1'
                   ip='192.0.2.1'
                   network='0'
                   port='3478'
                   priority='2130706431'
                   protocol='udp'
                   type='host'/>
      </transport>
    </content>
  </jingle>
</iq>
```

5.5 Candidate Negotiation

The initiator and responder negotiate connectivity over ICE by exchanging XML-formatted transport candidates for the channel. This negotiation proceeds immediately in order to maximize the possibility that connectivity can be established (and therefore media can be exchanged) as quickly as possible. In order to expedite session establishment, the initiator SHOULD include transport candidates in its session-initiate message but MAY also send additional transport candidates as soon as it learns of them, even before receiving the IQ-result that acknowledges the session-initiate message (i.e., the initiator MUST consider the session to be live as soon as it sends the session-initiate message). [21]

The first step in negotiating connectivity is for each party to send transport candidates to the other party. [23] These candidates SHOULD be gathered by following the procedure specified in Section 5.1.1 of RFC 8445 [25] (typically by communicating with a standalone STUN server in order to discover the client's public IP address and port) and prioritized by following the procedure specified in Section 5.1.2 of RFC 8445 [26].

Each candidate shall be sent as a <candidate/> child of a <transport/> element qualified by the 'urn:xmpp:jingle:transports:ice:0' namespace. The <transport/> element is sent via a Jingle message of type session-initiate, session-accept, or transport-info.

Either party MAY include multiple <candidate/> elements in one <transport/> element, especially in the session-initiate and session-accept messages sent at the beginning of the session negotiation. Including multiple candidates in the session-initiate and session-accept messages can help to ensure interoperability with entities that implement the SDP offer/answer model described in RFC 3264 [27]; in particular, an entity SHOULD include multiple candidates in its session-initiate or session-accept message if the other party advertises support for the "urn:ietf:rfc:3264" service discovery feature as described in the SDP Offer / Answer Support section of this document. However, including one candidate per subsequent transport-info message typically results in a faster negotiation because the candidates most likely to succeed are sent first (in the session-info and session-accept messages) and it is not necessary to gather all candidates before beginning to send any candidates; furthermore, because certain candidates can be more "expensive" in terms of bandwidth or processing power, either party might not want to advertise the existence of such candidates unless it is necessary to do so after other candidates have failed.

If the party that receives a candidate in a Jingle message can successfully process a given candidate or set of candidates, it returns an IQ-result (if not, for example because the candidate data is improperly formatted, it returns an IQ-error). At this point, the receiving entity is only indicating receipt of the candidate or set of candidates, not telling the other party that the candidate will be used.

The initiator can keep sending candidates (without stopping to receive an acknowledgement of receipt from the responder for each candidate) until it has exhausted its supply of possible or desirable transport candidates. The responder can also keep sending potential candidates, which the initiator will acknowledge.

[21] Given in-order delivery as mandated by XMPP Core [22], the responder will receive such transport-info messages after receiving the session-initiate message; if not, it is appropriate for the responder to return <unknown-session/> errors since according to its state machine the session does not exist.

[23] The fact that both parties send candidates means that Jingle requires each party to be a full implementation of ICE, not a lite implementation as specified in RFC 8445 [24].

[25] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[26] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[27] RFC 3264: An Offer/Answer Model with the Session Description Protocol (SDP) <http://tools.ietf.org/html/rfc3264>.

5.6 Connectivity Checks

As the initiator and responder receive candidates, they probe the candidates for connectivity. In performing these connectivity checks, each party SHOULD follow the procedure specified in Section 7 of RFC 8445 [28]. The following business rules apply:

1. Each party sends a STUN Binding Request (see RFC 5389 [29]) from each local candidate it generated to each remote candidate it received.
2. In accordance with RFC 8445 [30], the STUN Binding Requests MUST include the PRIORITY attribute (computed according to Section 7.1.2.1. of RFC 8445 [31]).
3. For the purposes of the Jingle ICE Transport Method, both parties are full ICE implementations and therefore the controlling role MUST be assumed by the initiator and the controlled role MUST be assumed by the responder.
4. The STUN Binding Requests generated by the initiator MAY include the USE-CANDIDATE attribute to indicate that the initiator wishes to cease checks for this component.
5. The STUN Binding Requests generated by the initiator MUST include the ICE-CONTROLLING attribute.
6. The STUN Binding Requests generated by the responder MUST include the ICE-CONTROLLED attribute.
7. The parties MUST use STUN short term credentials to authenticate requests and perform message integrity checks. As in RFC 8445 [32], the username in the STUN Binding Request is of the form ”ufrag-of-peer:ufrag-of-sender” and the password is the value of the ’pwd’ attribute provided by the peer. [33]

When it receives a STUN Binding Request, each party MUST return a STUN Binding Response, which indicates either an error case or the success case. As described in Section 7.2.5.3 of RFC 8445 [34], a connectivity check succeeds if *all* of the following are true:

[28] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[29] RFC 5389: Session Traversal Utilities for NAT (STUN) <http://tools.ietf.org/html/rfc5389>.

[30] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[31] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[32] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[33] Thus when Romeo sends a STUN Binding Request to Juliet the credentials will be STUN username ”9uB6:8hhy” (ufrag provided by Juliet concatenated with ufrag provided by Romeo) and password ”YH75Fviy6338Vbrhrlp8Yh” (pwd provided by Juliet) whereas when Juliet sends a STUN Binding Request to Romeo the credentials will be STUN username ”8hhy:9uB6” (ufrag provided by Romeo concatenated with ufrag provided by Juliet) and password ”asd88fgpdd777uzjYhagZg” (pwd provided by Romeo).

[34] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

1. The Binding request generated a success response.
2. The source and destination transport addresses in the Binding request and response are symmetric.

For the candidates exchanged in the previous section, the connectivity checks would be as follows (this diagram mirrors the example from section 15.1 of RFC 8445 [35]).

```
ENTITY                   IP Address  Mnemonic name
   --------------------------------------------------
   ICE Agent L (Initiator): 10.0.1.1    L-PRIV-1
   ICE Agent R (Responder): 192.0.2.1   R-PUB-1
   STUN Server:             192.0.2.2   STUN-PUB-1
   NAT (Public):            192.0.2.3   NAT-PUB-1


             L             NAT           STUN             R
             |STUN alloc.   |              |              |
             |(1) STUN Req  |              |              |
             |S=$L-PRIV-1   |              |              |
             |D=$STUN-PUB-1 |              |              |
             |------------->|              |              |
             |              |(2) STUN Req  |              |
             |              |S=$NAT-PUB-1  |              |
             |              |D=$STUN-PUB-1 |              |
             |              |------------->|              |
             |              |(3) STUN Res  |              |
             |              |S=$STUN-PUB-1 |              |
             |              |D=$NAT-PUB-1  |              |
             |              |MA=$NAT-PUB-1 |              |
             |              |<-------------|              |
             |(4) STUN Res  |              |              |
             |S=$STUN-PUB-1 |              |              |
             |D=$L-PRIV-1   |              |              |
             |MA=$NAT-PUB-1 |              |              |
             |<-------------|              |              |
             |(5) L’s␣Candidate␣Information|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣|------------------------------------------->|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣STUN
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣alloc.
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|(6)␣STUN␣Req␣␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|S=$R-PUB-1␣␣␣␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|D=$STUN-PUB-1␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|<-------------|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|(7)␣STUN␣Res␣␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|S=$STUN-PUB-1␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣|D=$R-PUB-1␣␣␣␣|
```

[35] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

```
␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|MA=$R-PUB-1␣␣␣|
␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|------------->|
␣␣␣␣␣␣␣␣␣␣␣␣␣␣|(8)␣R's Candidate Information|              |
              |<---------------------------------------------|
              |               |         (9) Bind Req         |Begin
              |               |         S=$R-PUB-1           |
                 Connectivity
              |               |         D=$L-PRIV-1          |Checks
              |               |         <--------------------|
              |               |         Dropped              |
              |(10) Bind Req  |               |              |
              |S=$L-PRIV-1    |               |              |
              |D=$R-PUB-1     |               |              |
              |------------->|               |              |
              |               |(11) Bind Req  |              |
              |               |S=$NAT-PUB-1   |              |
              |               |D=$R-PUB-1     |              |
              |               |----------------------------->|
              |               |(12) Bind Res  |              |
              |               |S=$R-PUB-1     |              |
              |               |D=$NAT-PUB-1   |              |
              |               |MA=$NAT-PUB-1  |              |
              |               |<-----------------------------|
              |(13) Bind Res  |               |              |
              |S=$R-PUB-1     |               |              |
              |D=$L-PRIV-1    |               |              |
              |MA=$NAT-PUB-1  |               |              |
              |<--------------|               |              |
              |Data           |               |              |
              |=============================================>|
              |               |               |              |
              |               |(14) Bind Req  |              |
              |               |S=$R-PUB-1     |              |
              |               |D=$NAT-PUB-1   |              |
              |               |<-----------------------------|
              |(15) Bind Req  |               |              |
              |S=$R-PUB-1     |               |              |
              |D=$L-PRIV-1    |               |              |
              |<--------------|               |              |
              |(16) Bind Res  |               |              |
              |S=$L-PRIV-1    |               |              |
              |D=$R-PUB-1     |               |              |
              |MA=$R-PUB-1    |               |              |
              |------------->|               |              |
              |               |(17) Bind Res  |              |
              |               |S=$NAT-PUB-1   |              |
              |               |D=$R-PUB-1     |              |
              |               |MA=$R-PUB-1    |              |
              |               |----------------------------->|
              |Data          |              |              |
              |<===========================================|
              |              |              |              |
                                  .......
              |              |              |              |
              |(18) Bind Req |              |              |
              |S=$L-PRIV-1   |              |              |
              |D=$R-PUB-1    |              |              |
              |USE-CAND      |              |              |
              |------------->|              |              |
              |              |(19) Bind Req |              |
              |              |S=$NAT-PUB-1  |              |
              |              |D=$R-PUB-1    |              |
              |              |USE-CAND      |              |
              |              |---------------------------->|
              |              |(20) Bind Res |              |
              |              |S=$R-PUB-1    |              |
              |              |D=$NAT-PUB-1  |              |
              |              |MA=$NAT-PUB-1 |              |
              |              |<----------------------------|
              |(21) Bind Res |              |              |
              |S=$R-PUB-1    |              |              |
              |D=$L-PRIV-1   |              |              |
              |MA=$NAT-PUB-1 |              |              |
              |<-------------|              |              |
              |              |              |              |
```

Note: aggressive nomination described in RFC 5245 is not used anymore in the updated RFC 8445 [36]. From now on the initiator MUST nominate just one valid candidate pair.

5.7 End-of-Candidates Indication

As explained in the Trickle ICE specification, when a party has completed gathering of ICE candidates it will send an "end-of-candidates indication" to the other party. In Jingle, this takes the form of an informational message as described under Informational Messages. This specificaton defines only a standalone "end-of-candidates indication" (i.e., not a way to indicate ICE completion in an offer or answer).

5.8 Acceptance of Successful Candidate

If, based on STUN connectivity checks, the parties determine that they will be able to exchange media (i.e., each component has "nominated" candidate pair and ICE processing is "completed"), they proceed with optional remote-candidate notification after which ICE transport is considered to be established. By this moment the parties may exchange media

[36] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

data already since it's allowed even before the candidate pairs nomination according to RFC 8445 [37]
Once the parties have connectivity and therefore the initiator has completed ICE for the media stream as explained in RFC 8445 [38], the initiator MAY communicate the in-use (nominated) candidate pairs in the signalling channel by sending a transport-info message that contains a <remote-candidate/> element for each component of the data stream (this maps to the SDP "remote-candidates" attribute as described in Appendix B of draft-ietf-mmusic-ice-sip-sdp specification, i.e., remote candidates are "the actual candidates at R that were selected by the offerer").
Note, while in SIP this message is MUST it's just MAY for XMPP. The difference comes from a SIP problem (offer updates) which doesn't exist in XMPP. Basically there is no **transport-info** or any other message which represents candidates of a valid pair and therefore the race condition is not possible. Even so if the responder advertises "urn:ietf:rfc:3264" disco feature and hence may serve as a Jingle-to-SIP proxy the message MUST be sent.

Listing 4: Initiator communicates in-use candidate

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='pd81b49s'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='set'>
 <jingle xmlns='urn:xmpp:jingle:1'
         action='transport-info'
         initiator='romeo@montague.example/dr4hcr0st3lup4c'
         sid='a73sjjvkla37jfea'>
   <content creator='initiator' name='this-is-the-audio-content'>
     <transport xmlns='urn:xmpp:jingle:transports:ice:0'
                pwd='asd88fgpdd777uzjYhagZg'
                ufrag='8hhy'>
       <remote-candidate component='1'
                         ip='10.0.1.2'
                         port='9001'/>
       <remote-candidate component='2'
                         ip='10.0.1.2'
                         port='9002'/>
     </transport>
   </content>
 </jingle>
</iq>
```

(In accordance with Jingle core, the responder will also acknowledge the transport-info message.)
In the unlikely event that one of the parties determines that it cannot establish connectivity even after sending and checking lower-priority candidates, it SHOULD terminate the session

[37] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.
[38] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

as described in Jingle (XEP-0166) [39], or alternatively it may do content-remove or transport-replace.

5.9 Negotiating a New Candidate

Even after media has begun to flow, either party MAY continue to send additional candidates to the other party (e.g., because the user agent has become aware of a new media proxy or network interface card). Such candidates are shared by sending a transport-info message.

Listing 5: Initiator sends a subsequent candidate

```
<iq from=’romeo@montague.example/dr4hcr0st3lup4c’
    id=’uh3g1f48’
    to=’juliet@capulet.example/yn0cl4bnw0yr3vym’
    type=’set’>
  <jingle xmlns=’urn:xmpp:jingle:1’
          action=’transport-info’
          initiator=’romeo@montague.example/dr4hcr0st3lup4c’
          sid=’a73sjjvkla37jfea’>
    <content creator=’initiator’ name=’this-is-the-audio-content’>
      <transport xmlns=’urn:xmpp:jingle:transports:ice:0’
                 pwd=’asd88fgpdd777uzjYhagZg’
                 ufrag=’8hhy’>
        <candidate component=’1’
                   foundation=’2B78DADC1A9E’
                   generation=’0’
                   id=’m3110wc4nd’
                   ip=’2001:db8::9:1’
                   network=’0’
                   port=’9001’
                   priority=’21149780477’
                   protocol=’udp’
                   type=’host’/>
      </transport>
    </content>
  </jingle>
</iq>
```

The receiving party MUST acknowledge receipt of the candidate.

Listing 6: Recipient acknowledges receipt

```
<iq from=’juliet@capulet.example/yn0cl4bnw0yr3vym’
    id=’uh3g1f48’
    to=’romeo@montague.example/dr4hcr0st3lup4c’
    type=’result’/>
```

[39] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.

The parties would check the newly-offered candidate for connectivity, as described previously. If the parties determine that media can flow over the candidate, they MAY then use the new candidate in subsequent communications.

5.10 ICE Restarts

At any time, either party MAY restart the process of ICE negotiation by sending a candidate with a ’generation’ value that is greater than the previous generation of candidates; when it does so, it MUST generate new values for the ’pwd’ and ’ufrag’ attributes, consistent with the definition of an ICE restart in Section 9 of RFC 8445 [40] (because an ICE restart is signalled by a change in the ’pwd’ and ’ufrag’ attributes, strictly speaking the ’generation’ attribute is not absolutely necessary). As explained in RFC 8445 [41], typically the ICE negotiation would be restarted to change the media target (e.g., an IP address change for one of the parties) and certain third-party-call-control scenarios.

Listing 7: Initiator restarts ICE negotiation

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='kl23fs71'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-info'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='this-is-the-audio-content'>
      <transport xmlns='urn:xmpp:jingle:transports:ice:0'
                 pwd='bv71hdn38hgb39hf6xlk33'
                 ufrag='g7qs'>
        <candidate component='1'
                   foundation='2B78DADC1A9E'
                   generation='1'
                   id='y3s2b30v3r'
                   ip='192.0.2.3'
                   network='1'
                   port='45665'
                   priority='1694498815'
                   protocol='udp'
                   type='srflx'/>
      </transport>
    </content>
  </jingle>
</iq>
```

[40] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

[41] RFC 8445: Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc8445>.

The recipient then acknowledges receipt.

Listing 8: Recipient acknowledges transport-info

```
<iq from='juliet@capulet.example/yn0cl4bnw0yr3vym'
    id='kl23fs71'
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='result'/>
```

The parties would then exchange new candidates to renegotiate connectivity and would check the new candidates for connectivity, as described previously. If the parties determine that media can flow over one of the new candidates, they can then use the successful candidate in subsequent communications. However, while ICE is being renegotiated the parties can continue to send media with the existing candidate-in-use.
Note: If a party has already sent ICE restart and receives any transport-info message before <iq/> stanza of type "result", the transport-info messages have to be acknowledged with <iq/> stanzas of type "result" but dropped afterwards. After the restart was acknowledged, the other party MAY send the same candidates again as a part of the new ICE session. It's also possible both parties will send ICE restart simultaneously. In this case session initiator MUST respond with <tie-break/> error (see Jingle (XEP-0166) [42]).

6 Fallback to Raw UDP

It can happen that the responder does not support ICE, in which case it can be necessary to fall back to use of the Jingle Raw UDP Transport Method (XEP-0177) [43]. One typical scenario is communication between an ICE-aware Jingle endpoint and a non-ICE-aware SIP endpoint through a Jingle-to-SIP gateway, as follows:

1. The Jingle endpoint sends a session-initiate request to the SIP endpoint, specifying a transport method of ICE.
2. Based on capabilities information, the gateway knows that the SIP endpoint does not support ICE, so it enables the endpoints to use its media relay. It does this by:
    - Sending a transport-replace message to the Jingle endpoint on behalf of the SIP endpoint, specifying a transport method of Raw UDP and a candidate whose IP address and port are hosted at the gateway.
    - Sending SIP INVITE to the SIP endpoint on behalf of the Jingle endpoint, speciying an IP address and port at the gateway.

The session flow is as follows.

[42] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.
[43] XEP-0177: Jingle Raw UDP Transport Method <https://xmpp.org/extensions/xep-0177.html>.

```
Romeo                    Gateway                    Juliet
  |                         |                         |
  |   session-initiate      |                         |
  |   (audio definition)    |                         |
  |------------------------>|                         |
  |   ack                   |                         |
  |<------------------------|                         |
  |   transport-replace     |                         |
  |   (Raw UDP)             |                         |
  |<------------------------|                         |
  |   ack                   |                         |
  |------------------------>|                         |
  |   transport-accept      |                         |
  |------------------------>|                         |
  |   ack                   |                         |
  |<------------------------|   SIP INVITE            |
  |                         |------------------------>|
  |                         |   200 OK                |
  |                         |<------------------------|
  |   session-accept        |                         |
  |<------------------------|                         |
  |   ack                   |                         |
  |------------------------>|                         |
  |                    MEDIA SESSION                  |
  |<=================================================>|
  |                         |   SIP BYE               |
  |                         |<------------------------|
  |   session-terminate     |                         |
  |<------------------------|                         |
  |   ack                   |                         |
  |------------------------>|   ack                   |
  |                         |------------------------>|
  |                         |                         |
```

The protocol flow is as follows, showing only the stanzas sent between Romeo and the gateway (acting on Juliet's behalf).

Listing 9: Initiator sends session-initiate

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='p01hf63x'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='session-initiate'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='voice'>
      <description xmlns='urn:xmpp:jingle:apps:rtp:1' media='audio'>
```

```
        <payload-type id='96' name='speex' clockrate='16000'/>
        <payload-type id='97' name='speex' clockrate='8000'/>
        <payload-type id='18' name='G729'/>
        <payload-type id='103' name='L16' clockrate='16000' channels='
           2'/>
        <payload-type id='98' name='x-ISAC' clockrate='8000'/>
      </description>
      <transport xmlns='urn:xmpp:jingle:transports:ice:0'
                 pwd='asd88fgpdd777uzjYhagZg'
                 ufrag='8hhy'
                 ice2='true'>
        <candidate component='1'
                   foundation='2B78DADC1A9E'
                   generation='0'
                   id='el0747fg11'
                   ip='10.0.1.1'
                   network='1'
                   port='8998'
                   priority='2130706431'
                   protocol='udp'
                   type='host'/>
        <candidate component='1'
                   foundation='58AA96B8FA5A'
                   generation='0'
                   id='y3s2b30v3r'
                   ip='192.0.2.3'
                   network='1'
                   port='45664'
                   priority='1694498815'
                   protocol='udp'
                   rel-addr='10.0.1.1'
                   rel-port='8998'
                   type='srflx'/>
      </transport>
    </content>
  </jingle>
</iq>
```

Listing 10: Responder acknowledges session-initiate

```
<iq from='juliet@capulet.example/yn0cl4bnw0yr3vym'
    id='p01hf63x'
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='result'/>
```

Immediately the gateway sends a transport-replace message to Romeo, specifying a transport of Raw UDP with a candidate whose IP address and port identify a media relay at the gateway.

Listing 11: Gateway sends transport-replace on behalf of responder

```
<iq from='juliet@capulet.example/yn0cl4bnw0yr3vym'
    id='hy2gd714'
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-replace'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='voice1'>
      <transport xmlns='urn:xmpp:jingle:transports:raw-udp:1'>
        <candidate generation='0'
                   id='a9j3mnbtu1'
                   ip='10.1.1.104'
                   port='13540'/>
      </transport>
    </content>
  </jingle>
</iq>
```

Romeo then acknowledges the transport-replace message and immediately also sends a transport-accept.

Listing 12: Initiator acknowledges transport-replace

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='hy2gd714'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='result'/>
```

Listing 13: Initiator accepts new transport

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='rb391gs5'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-accept'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          sid='a73sjjvkla37jfea'>
    <content creator='responder' name='voice2'>
      <transport xmlns='urn:xmpp:jingle:transports:raw-udp:1'>
        <candidate generation='0'
                   id='a9j3mnbtu1'
                   ip='10.1.1.104'
                   port='13540'/>
      </transport>
    </content>
  </jingle>
</iq>
```

The gateway then acknowledges the acceptance on behalf of Juliet.

Listing 14: Gateway acknowledges transport-accept

```
<iq from=’juliet@capulet.example/yn0cl4bnw0yr3vym’
    id=’rb391gs5’
    to=’romeo@montague.example/dr4hcr0st3lup4c’
    type=’result’/>
```

The responder then sends a session-accept through the gateway.

Listing 15: Responder sends session-accept

```
<iq from=’juliet@capulet.example/yn0cl4bnw0yr3vym’
    id=’ijf61d43’
    to=’romeo@montague.example/dr4hcr0st3lup4c’
    type=’set’>
  <jingle xmlns=’urn:xmpp:jingle:1’
          action=’session-accept’
          initiator=’romeo@montague.example/dr4hcr0st3lup4c’
          responder=’juliet@capulet.example/yn0cl4bnw0yr3vym’
          sid=’a73sjjvkla37jfea’>
    <content creator=’initiator’ name=’voice’>
      <description xmlns=’urn:xmpp:jingle:apps:rtp:1’ media=’audio’>
        <payload-type id=’18’ name=’G729’/>
      </description>
      <transport xmlns=’urn:xmpp:jingle:transports:raw-udp:1’/>
    </content>
  </jingle>
</iq>
```

Listing 16: Initiator acknowledges session-accept

```
<iq from=’romeo@montague.example/dr4hcr0st3lup4c’
    id=’ijf61d43’
    to=’juliet@capulet.example/yn0cl4bnw0yr3vym’
    type=’result’/>
```

The endpoints now begin to exchange session media, and can continue the session as long as desired.

7 Informational Messages

Informational messages can be sent by either party within the context of Jingle to communicate the status of a Jingle ICE ”session”. The informational message MUST be an IQ-set containing a <jingle/> element of type ”transport-info”, where the informational message is

a payload element qualified by the 'urn:xmpp:jingle:transports:ice:0' namespace.
The only payload element defined so far is the <gathering-complete/> element. This element is used only to signal that gathering of ICE candidates has been completed (i.e., to send an "end-of-candidates indication"), as in the following example.

Listing 17: Responder sends end-of-candidates indication

```
<iq from='juliet@capulet.example/yn0cl4bnw0yr3vym'
    id='xv39z423'
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-info'
          initiator='romeo@montague.example/dr4hcr0st3lup4c'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='this-is-the-audio-content'>
      <transport xmlns='urn:xmpp:jingle:transports:ice:0'
                 pwd='asd88fgpdd777uzjYhagZg'
                 ufrag='8hhy'>
        <gathering-complete/>
      </transport>
    </content>
  </jingle>
</iq>
```

The <gathering-complete/> element can be combined with remaining candidates or sent alone.

8 Determining Support

8.1 ICE Support

To advertise its support for the Jingle ICE Transport Method, when replying to Service Discovery (XEP-0030) [44] information requests an entity MUST return URNs for any version of this protocol that the entity supports -- e.g., "urn:xmpp:jingle:transports:ice:0" for this version (and "urn:xmpp:jingle:transports:ice-udp:1" for the "ICE-UDP" version previously specified in XEP-0176 (see Namespace Versioning regarding the possibility of incrementing the version number).

Listing 18: Service discovery information request

```
<iq from='romeo@montague.example/dr4hcr0st3lup4c'
    id='cv5x41g9'
    to='juliet@capulet.example/yn0cl4bnw0yr3vym'
    type='get'>
```

[44] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

```
  <query xmlns=’http://jabber.org/protocol/disco#info’/>
</iq>
```

Listing 19: Service discovery information response

```
<iq from=’juliet@capulet.example/yn0cl4bnw0yr3vym’
    id=’cv5x41g9’
    to=’romeo@montague.example/dr4hcr0st3lup4c’
    type=’result’>
  <query xmlns=’http://jabber.org/protocol/disco#info’>
    <feature var=’urn:xmpp:jingle:1’/>
    <feature var=’urn:xmpp:jingle:transports:ice:0’/>
    <feature var=’urn:xmpp:jingle:transports:ice-udp:1’/>
    <feature var=’urn:xmpp:jingle:apps:rtp:1’/>
    <feature var=’urn:xmpp:jingle:apps:rtp:audio’/>
    <feature var=’urn:xmpp:jingle:apps:rtp:video’/>
  </query>
</iq>
```

In order for an application to determine whether an entity supports this protocol, where possible it SHOULD use the dynamic, presence-based profile of service discovery defined in Entity Capabilities (XEP-0115) [45]. However, if an application has not received entity capabilities information from an entity, it SHOULD use explicit service discovery instead.

8.2 SDP Offer / Answer Support

If an entity supports the SDP offer / answer model described in RFC 3264 [46] and therefore prefers to receive multiple candidates in a single transport-info message, it MUST advertise support for the ”urn:ietf:rfc:3264” service discovery feature. Typically this feature will be advertised only by gateways between Jingle and SIP.

Listing 20: Service discovery information request

```
<iq from=’romeo@montague.example/dr4hcr0st3lup4c’
    id=’ce81f5d6’
    to=’sip.shakespeare.lit’
    type=’get’>
  <query xmlns=’http://jabber.org/protocol/disco#info’/>
</iq>
```

Listing 21: Service discovery information response

```
<iq from=’sip.shakespeare.lit’
    id=’ce81f5d6’
```

[45] XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.

[46] RFC 3264: An Offer/Answer Model with the Session Description Protocol (SDP) <http://tools.ietf.org/html/rfc3264>.

```
    to='romeo@montague.example/dr4hcr0st3lup4c'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <feature var='urn:ietf:rfc:3264'/>
    <feature var='urn:xmpp:jingle:1'/>
    <feature var='urn:xmpp:jingle:transports:ice:0'/>
    <feature var='urn:xmpp:jingle:transports:ice-udp:1'/>
    <feature var='urn:xmpp:jingle:apps:rtp:1'/>
    <feature var='urn:xmpp:jingle:apps:rtp:audio'/>
    <feature var='urn:xmpp:jingle:apps:rtp:video'/>
  </query>
</iq>
```

9 Implementation Notes

In order to speed the negotiation process so that media can flow as quickly as possible, the initiator SHOULD gather and prioritize candidates in advance, or as soon as the principal begins the process of initiating a session.

10 Deployment Notes

This specification applies exclusively to Jingle clients and places no additional requirements on XMPP servers. However, service administrators might wish to deploy a STUN server in order to ease the client-to-client negotiation process and a TURN server for media relaying (see TURN [47]). Deployment of support for External Service Discovery (XEP-0215) [48] might also be helpful.

11 Security Considerations

11.1 Sharing IP Addresses

By definition, the exchange of transport candidates results in exposure of the sender's IP addresses, which comprise a form of personally identifying information. A Jingle client MUST enable a user to control which entities will be allowed to receive such information. If a human user explicitly accepts a session request, then the client SHOULD consider that action to imply approval of IP address sharing. However, waiting for a human user to explicitly accept the session request can result in delays during session setup, since it is more efficient to immediately begin sharing transport candidates. Therefore, it is RECOMMENDED for the

[47] Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN) <http://tools.ietf.org/html/draft-ietf-behave-turn>. Work in progress.

[48] XEP-0215: External Service Discovery <https://xmpp.org/extensions/xep-0215.html>.

client to immediately send transport candidates to a contact (without waiting for explicit user approval of the session request) in the following cases:

1. The user has permanently and formally authorized the contact to view the user's presence information via a presence subscription as reflected in an XMPP roster item (see XMPP IM [49]).
2. The user has temporarily and dynamically shared presence with the contact via "directed presence" as described in RFC 3921 [50].
3. The user has explicitly added the contact to a list of entities who are allowed to access the user's personally-identifying information.

11.2 Encryption of Media

A Jingle implementation SHOULD support security preconditions that are enforced before application media is allowed to flow over a UDP association, such as those described in Jingle XTLS [51].
Application types that use the Jingle ICE transport method MAY also define their own application-specific encryption methods, such as the Secure Real-time Transport Protocol (SRTP) for RTP exchanges as described in Jingle RTP Sessions (XEP-0167) [52].

12 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [53].

13 XMPP Registrar Considerations

13.1 Protocol Namespaces

This specification defines the following XML namespace:

[49] RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc6121>.

[50] RFC 3921: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc3921>.

[51] Extensible Messaging and Presence Protocol (XMPP) End-to-End Encryption Using Transport Layer Security ("XTLS") <http://tools.ietf.org/html/draft-meyer-xmpp-e2e-encryption>.

[52] XEP-0167: Jingle RTP Sessions <https://xmpp.org/extensions/xep-0167.html>.

[53] The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <http://www.iana.org/>.

- urn:xmpp:jingle:transports:ice:0

The XMPP Registrar [54] includes the foregoing namespace in its registry at <https://xmpp.org/registrar/namespaces.html>, as governed by XMPP Registrar Function (XEP-0053) [55].

13.2 Protocol Versioning

If the protocol defined in this specification undergoes a revision that is not fully backwards-compatible with an older version, the XMPP Registrar shall increment the protocol version number found at the end of the XML namespaces defined herein, as described in Section 4 of XEP-0053.

13.3 Service Discovery Features

If an entity supports the SDP offer / answer model described in RFC 3264 [56] and therefore prefers to receive one transport-info message with multiple candidates, it MUST advertise support for the "urn:ietf:rfc:3264" feature.
The registry submission is as follows.

```
<var>
  <name>urn:ietf:rfc:3264</name>
  <desc>
    Signals support for the SDP offer / answer model
    described in RFC 3264
  </desc>
  <doc>XEP-0176</doc>
</var>
```

13.4 Jingle Transport Methods

The XMPP Registrar includes "ice" in its registry of Jingle transport methods at <https://xmpp.org/registrar/jingle-transports.html>. The registry submission is as follows:

```
<transport>
  <name>ice</name>
```

[54] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

[55] XEP-0053: XMPP Registrar Function <https://xmpp.org/extensions/xep-0053.html>.

[56] RFC 3264: An Offer/Answer Model with the Session Description Protocol (SDP) <http://tools.ietf.org/html/rfc3264>.

```
  <desc>
    A method for negotiation of out-of-band UDP associations
    or TCP connections with built-in NAT and firewall traversal
    using the IETF's␣Interactive␣Connectivity␣Establishment␣(ICE)
␣␣␣␣methodology.
␣␣</desc>
␣␣<type>datagram␣or␣streaming</type>
␣␣<doc>XEP-0176</doc>
</transport>
```

14 XML Schema

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:jingle:transports:ice:0'
    xmlns='urn:xmpp:jingle:transports:ice:0'
    elementFormDefault='qualified'>

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0176: http://www.xmpp.org/extensions/xep-0176.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='transport'>
    <xs:complexType>
      <xs:choice minOccurs='0'>
        <xs:sequence>
          <xs:element name='candidate'
                      type='candidateElementType'
                      minOccurs='1'
                      maxOccurs='unbounded'/>
        </xs:sequence>
        <xs:sequence>
          <xs:element name='remote-candidate'
                      type='remoteCandidateElementType'
                      minOccurs='1'
                      maxOccurs='unbounded'/>
        </xs:sequence>
        <xs:sequence>
          <xs:any namespace="##other"
                  processContents="lax"
                  minOccurs="0"
                  maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:choice>
    <xs:attribute name='pwd' type='xs:string' use='optional'/>
    <xs:attribute name='ufrag' type='xs:string' use='optional'/>
    <xs:attribute name='ice2' type='xs:boolean' use='optional'
       default='false'/>
  </xs:complexType>
</xs:element>

<xs:complexType name='candidateElementType'>
  <xs:simpleContent>
    <xs:extension base='empty'>
      <xs:attribute name='component' type='xs:unsignedByte' use='
         required'/>
      <xs:attribute name='foundation' type='xs:string' use='required
         '/>
      <xs:attribute name='generation' type='xs:unsignedByte' use='
         optional'/>
      <xs:attribute name='id' type='xs:NCName' use='optional'/>
      <xs:attribute name='ip' type='xs:string' use='required'/>
      <xs:attribute name='network' type='xs:unsignedByte' use='
         required'/>
      <xs:attribute name='port' type='xs:unsignedShort' use='
         required'/>
      <xs:attribute name='priority' type='xs:positiveInteger' use='
         required'/>
      <xs:attribute name='protocol' type='xs:NCName' use='required'/
         >
      <xs:attribute name='rel-addr' type='xs:string' use='optional'/
         >
      <xs:attribute name='rel-port' type='xs:unsignedShort' use='
         optional'/>
      <xs:attribute name='tcptype' use='optional'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='active'/>
            <xs:enumeration value='passive'/>
            <xs:enumeration value='so'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name='type' use='required'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='host'/>
            <xs:enumeration value='prflx'/>
            <xs:enumeration value='relay'/>
            <xs:enumeration value='srflx'/>
          </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name=’remoteCandidateElementType’>
    <xs:simpleContent>
      <xs:extension base=’empty’>
        <xs:attribute name=’component’ type=’xs:unsignedByte’ use=’
            required’/>
        <xs:attribute name=’ip’ type=’xs:string’ use=’required’/>
        <xs:attribute name=’port’ type=’xs:unsignedShort’ use=’
            required’/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:element name=’gathering-complete’ type=’empty’/>

  <xs:simpleType name=’empty’>
    <xs:restriction base=’xs:string’>
      <xs:enumeration value=’’/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

15 Acknowledgements

Special thanks to Joe Beda, Scott Ludwig, Joe Hildebrand, Sean Egan, and Robert McQueen for co-authoring XEP-0176, from which this document was forked.
Thanks also to Diana Cionoiu, Olivier Crête, Philipp Hancke, Tim Julien, Steffen Larsen, Unnikrishnan Vikrama Panicker, Mike Ruprecht, Lance Stout, Justin Uberti, and Paul Witty for their feedback.