XEP-0260: Jingle SOCKS5 Bytestreams Transport Method

Peter Saint-Andre
mailto:peter@andyet.net
xmpp:stpeter@stpeter.im
https://stpeter.im/

Dirk Meyer
mailto:dmeyer@tzi.de
xmpp:dmeyer@jabber.org

Justin Karneges
mailto:justin@affinix.com
xmpp:justin@andbit.net

Marcus Lundblad
mailto:ml@update.uu.se
xmpp:mlundblad@jabber.org

Tobias Markmann
mailto:tobias.markmann@isode.com
xmpp:tm@ayena.de

Klaus Hartke
mailto:klaus.hartke@googlemail.com
xmpp:nx@jabber.org

2016-05-17
Version 1.0.1

Status	Type	Short Name
Draft	Standards Track	jingle-s5b

This specification defines a Jingle transport method that results in sending data via the SOCKS5 Bytestreams (S5B) protocol defined in XEP-0065. Essentially this transport method reuses XEP-0065 semantics for sending the data and defines native Jingle methods for starting and ending an S5B session.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

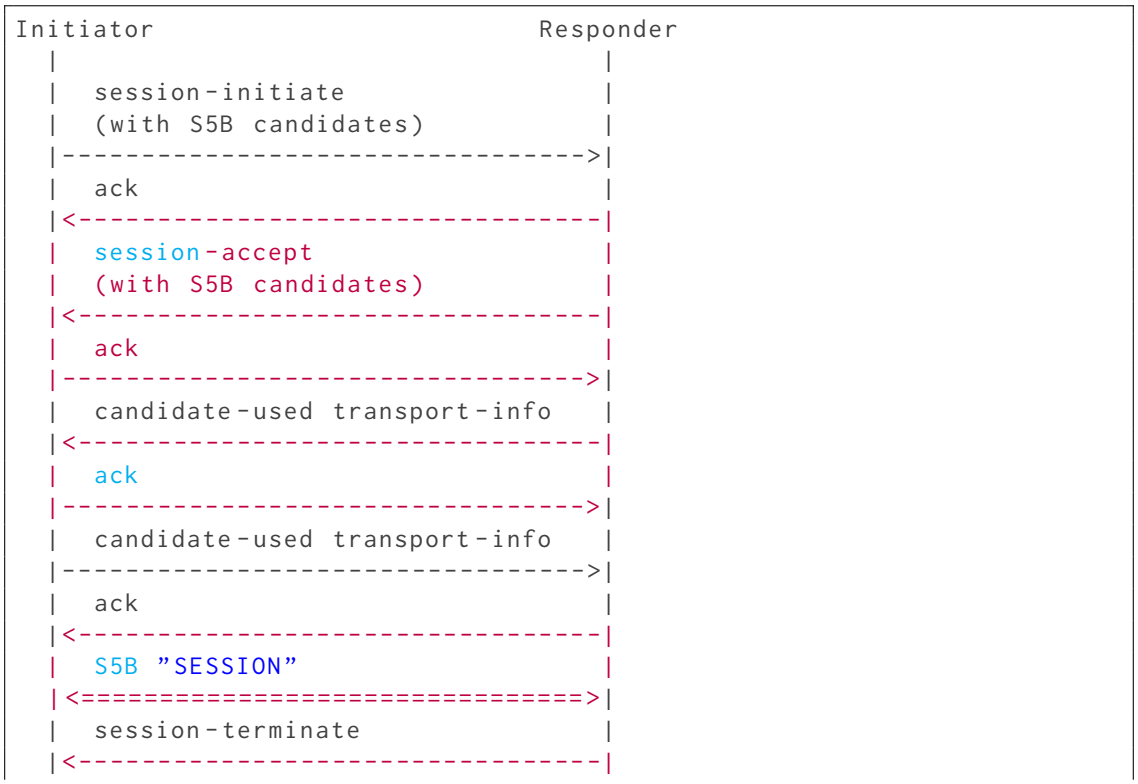
Contents

1 Introduction

Jingle (XEP-0166) [1] defines a framework for negotiating and managing 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 using the existing SOCKS5 Bytestreams (S5B) protocol specified in SOCKS5 Bytestreams (XEP-0065) [2]. This "jingle-s5b" method results in a streaming transport method suitable for use in Jingle application types where packet loss cannot be tolerated (e.g., file transfer). Jingle-S5B reuses the protocol flow from XEP-0065 for the communication with a SOCKS5 streamhost; the communication between two clients to negotiate the possible candidates differs from XEP-0065 and shares similarities with Jingle ICE-UDP Transport Method (XEP-0176) [3]

2 Protocol

The basic flow is as follows.



[1] XEP-0166: Jingle <https://xmpp.org/extensions/xep-0166.html>.
[2] XEP-0065: SOCKS5 Bytestreams <https://xmpp.org/extensions/xep-0065.html>.
[3] XEP-0176: Jingle ICE-UDP Transport Method <https://xmpp.org/extensions/xep-0176.html>.

This flow is illustrated in the following examples (to simplify the presentation these use an "example" application instead of a real application type).

2.1 Selecting Candidates

It is RECOMMENDED that a client will offer as many <candidate/> elements as possible with itself as the host (i.e., non-proxy candidates). Such candidates might be found using the following methods:

- Opening the TCP port on all available interfaces the user wants to use (e.g., maybe not an expensive UMTS link), including the IPv4 and IPv6 addresses of that interface (if available).
- Using the client's external IP address as discovered through an assisting NAT protocol or other means.

If the client knows it is behind a NAT and the router announces UPnP IGD or NAT-PMP support, the client SHOULD map the open port to the external interface of the router and include the public IP address and port information in the <candidate/> offers. To increase the chance of success without using a proxy, this specification encourages the responder to also send offers, effectively equivalent to the "fast-mode" for SOCKS5 Bytestreams as previously described at <http://delta.affinix.com/specs/stream.html>.

2.2 Exchanging Candidates

Once the initiator has a set of candidates, it sends a Jingle session-initiate request that contains one or more transport candidates which are a mixture of XEP-0065 streamhosts and ICE candidates used in XEP-0176.
Just as with the <query/> element from XEP-0065, here the <transport/> element contains the candidates. The following rules apply to the defined attributes of the <transport/> element when sent by the initiator in a Jingle session-initate message:

1. The 'sid' attribute MUST be included. This attribute specifies the Stream ID for this bytestream.
2. The 'dstaddr' attribute SHOULD be included if the initiator includes at least one candidate of the "proxy" type. This attribute enables the initiator to communicate the value it has calculated for the SOCKS5 DST.ADDR field (see Section 5.3.2 and Section 7 of XEP-0065) so that the responder can provide an accurate value to the proxy during SOCKS5

negotiation. Here the value is calculated as SHA1(SID + Initiator JID + Responder JID) since the initiator will be the entity that activates the bytestream at the proxy. [4]

3. The 'mode' attribute MAY be included. This attribute specifies whether the underlying transport for the bytestream will be TCP (a value of "tcp", which is the default) or UDP (a value of "udp", see Section 8 of XEP-0065).

In the following example, Romeo's client has two interfaces, one on port 5086 and the other on port 5087. The provided candidates are the IPv4 address of one interface, the IPv4 address of the second interface, and a proxy address at streamer.shakespeare.lit. Because Romeo's client has included a proxy candidate, it includes its computed value for the DST.ADDR field in the 'dstaddr' attribute (here computed as the SHA-1 hash of "vj3hs98yromeo@montague.lit/orchardjuliet@capulet.lit/balcony").

Listing 1: Initiator sends session-initiate

```
<iq from='romeo@montague.lit/orchard'
    id='xn28s7gk'
    to='juliet@capulet.lit/balcony'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='session-initiate'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <description xmlns='urn:xmpp:example'/>
      <transport xmlns='urn:xmpp:jingle:transports:s5b:1'
                 dstaddr='972b7bf47291ca609517f67f86b5081086052dad'
                 mode='tcp'
                 sid='vj3hs98y'>
        <candidate cid='hft54dqy'
                   host='192.168.4.1'
                   jid='romeo@montague.lit/orchard'
                   port='5086'
                   priority='8257636'
                   type='direct'/>
        <candidate cid='hutr46fe'
                   host='24.24.24.1'
                   jid='romeo@montague.lit/orchard'
                   port='5087'
                   priority='8258636'
```

[4] In XEP-0065, the DST.ADDR is always calculated as SHA1(SID + Requester JID + Target JID); in XEP-0260 the Jingle "initiator" is the SOCKS5 Bytestreams "requester" and the Jingle "responder" is the SOCKS5 Bytestreams "target", so for proxy candidates sent from the initiator/requester to the responder/target the DST.ADDR is calculated as SHA1(SID + Initiator JID + Responder JID). Note well that the calcuation for proxy candidates sent from the responder/target to the initiator/requester is SHA1(SID + Responder JID + Initiator JID); this scenario is not covered by XEP-0065 since in that specification only the SOCKS5 Bytestreams "requester" provides candidates.

```
                    type='direct'/>
        <candidate  cid='xmdh4b7i'
                    host='123.456.7.8'
                    jid='streamer.shakespeare.lit'
                    port='7625'
                    priority='7878787'
                    type='proxy'/>
      </transport>
    </content>
  </jingle>
</iq>
```

The responder immediately acknowledges receipt.

Listing 2: Responder acknowledges session-initiate

```
<iq from='juliet@capulet.lit/balcony'
    id='xn28s7gk'
    to='romeo@montague.lit/orchard'
    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 Jingle session-accept message.
This message MUST contain a <transport/> element qualified by the 'urn:xmpp:jingle:transports:s5b:1' namespace, which SHOULD in turn contain one <candidate/> element for each SOCKS5 Bytestreams candidate generated by or known to the responder, but MAY instead be empty if the responder does not wish to offer any candidates or wishes to send each candidate as the payload of a transport-info message. If the responder sends candidates in the session-accept, the chances of a successful connection are increased. For example, the initiator might be behind a NAT or might have no access to an S5B proxy, whereas the responder might have a public IP address, might know about a proxy, or might have NAT penetration support like NAT-PMP in a router. However, the responder MUST NOT offer as a candidate any host/port combination that has already been offered by the initiator; this helps to prevent failure of negotiation with S5B proxies.
The following rules apply to the defined attributes of the <transport/> element when sent by the responder in a Jingle session-accept message:

1. The 'sid' attribute MUST be included and MUST be the same Stream ID communicated by the initiator in the Jingle session-initiate message.

2. The 'dstaddr' attribute SHOULD be included if the responder includes at least one candidate of the "proxy" type. This attribute enables the responder to communicate the value

it has calculated for the SOCKS5 DST.ADDR field (see Section 5.3.2 and Section 7 of XEP-0065) so that the initiator can provide an accurate value to the proxy during SOCKS5 negotiation. Here the value is calculated as SHA1(SID + Responder JID + Initiator JID) since the responder will be the entity that activates the bytestream at the proxy. [5]

3. The 'mode' attribute MUST NOT be included since the underlying transport for the bytestream is determined by the initiator.

In the following example, Juliet's client opens one port. The provided candidates are the (private) IPv4 address of the interface, a (public) IPv6 address, the public IPv4 address created by mapping the private IP address/port using NAT-PMP, and a proxy address. Because Juliet's client has included a proxy candidate, it includes its computed value for the DST.ADDR field in the 'dstaddr' attribute (here computed as the SHA-1 hash of "vj3hs98yjuliet@capulet.lit/balconyromeo@montague.lit/orchard").

Listing 3: Responder sends session-accept with candidates

```
<iq from='juliet@capulet.lit/balcony'
    id='hwd987h'
    to='romeo@montague.lit/orchard'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='session-accept'
          responder='juliet@capulet.lit/balcony'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <description xmlns='urn:xmpp:example'/>
      <transport xmlns='urn:xmpp:jingle:transports:s5b:1'
                 dstaddr='1a12fb7bc625e55f3ed5b29a53dbe0e4aa7d80ba'
                 mode='tcp'
                 sid='vj3hs98y'>
        <candidate cid='ht567dq'
                   host='192.169.1.10'
                   jid='juliet@capulet.lit/balcony'
                   port='6539'
                   priority='8257636'
                   type='direct'/>
        <candidate cid='grt654q2'
                   host='2001:638:708:30c9:219:d1ff:fea4:a17d'
                   jid='juliet@capulet.lit/balcony'
                   port='6539'
                   priority='8257606'
                   type='direct'/>
        <candidate cid='hr65dqyd'
                   host='134.102.201.180'
```

[5] As noted, the calculation for proxy candidates sent from the responder/target to the initiator/requester is SHA1(SID + Responder JID + Initiator JID); this scenario is not covered by XEP-0065 since in that specification only the SOCKS5 Bytestreams "requester" provides candidates.

```
                    jid='juliet@capulet.lit/balcony'
                    port='16453'
                    priority='7929856'
                    type='assisted'/>
          <candidate cid='pzv14s74'
                    host='234.567.8.9'
                    jid='proxy.marlowe.lit'
                    port='7676'
                    priority='7788877'
                    type='proxy'/>
      </transport>
    </content>
  </jingle>
</iq>
```

The initiator acknowledges receipt and tries to connect to the offered candidates.

Listing 4: Initiator acknowledges session-accept

```
<iq from='romeo@montague.lit/orchard'
    id='hwd987h'
    to='juliet@capulet.lit/balcony'
    type='result'/>
```

A client SHOULD check the offered candidates in order of their priority, starting with the highest value. How the priority is calculated depends on the actual available interfaces. An implementation SHOULD use the following formula:

```
priority = (2^16)*(type preference) + (local preference)
```

The type preference is an integer value between 0 and 127. The following types and their suggested preference values are defined.

Type	Description	Preference Value
direct	Direct connection using the given interface	126
assisted	Direct connection using NAT assisting technologies like NAT-PMP or UPnP-IGD	120
tunnel	Tunnel protocols such as Teredo	110
proxy	SOCKS5 Relay	10

The local preference is used to rate different candidates of the same type, e.g. a DSL link might be preferred over a VPN connection. The value of the local preference SHOULD be

between 0 and 65535. The proposed values are only guidelines. If a client wants to increase or decrease the value of a specific candidate it is free to do so. For instance, a client might have an expensive UMTS link as a last resort and might rate this link lower than all SOCKS5 relays.

2.3 Connecting to Candidates

After receiving its peer's candidates, a client start to connect to them in order of the priority. A detailed description of the protocol can be found in XEP-0065.
Once one client has successfully created a connection, it sends the <candidate-used/> element to the peer inside a Jingle transport-info message. If a client receives a candidate-used notification it SHOULD continue trying to connect to candidates sent by its peer if it has not tried all candidates with a higher priority than the one successfully used by the peer.

Listing 5: Initiator sends candidate-used in Jingle transport-info

```
<iq from='romeo@montague.lit/orchard'
    id='hjdi8'
    to='juliet@capulet.lit/balcony'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-info'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <transport xmlns='urn:xmpp:jingle:transports:s5b:1'
                 sid='vj3hs98y'>
        <candidate-used cid='hr65dqyd'/>
      </transport>
    </content>
  </jingle>
</iq>
```

The peer immediately acknowledges receipt.

Listing 6: Responder acknowledges candidate-used message

```
<iq from='juliet@capulet.lit/balcony'
    id='hjdi8'
    to='romeo@montague.lit/orchard'
    type='result'/>
```

If a client is unable to connect to *any* candidate sent by its peer, or if it stopped trying to connect because its peer sent a candidate-used notification with a priority higher than its remaining candidate(s), it sends a candidate-error Jingle transport-info message (this is equivalent to the IQ-error with code='500' from the "fast-mode" extension).

Listing 7: Responder sends candidate-error in Jingle transport-info

```
<iq from='juliet@capulet.lit/balcony'
    id='gft65ewd'
    to='romeo@montague.lit/orchard'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-info'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <transport xmlns='urn:xmpp:jingle:transports:s5b:1'
                 sid='vj3hs98y'>
        <candidate-error/>
      </transport>
    </content>
  </jingle>
</iq>
```

The peer immediately acknowledges receipt.

Listing 8: Responder acknowledges candidate-error message

```
<iq from='romeo@montague.lit/orchard'
    id='gft65ewd'
    to='juliet@capulet.lit/balcony'
    type='result'/>
```

2.4 Completing the Negotiation

The transport negotiation is completed in one of the following ways:

1. If both parties send a candidate-error notification then the SOCKS5 negotiation has failed and the parties need to fall back to some other transport method, typically (but not necessarily) IBB; see the Fallback Methods section of this document for details.
2. If one of the parties sends a candidate-error notification and the other party sends a candidate-used notification, then the candidate-used shall be considered the nominated candidate.
3. If both parties send a candidate-used notification but the candidates have a different priority, then the candidate with the higher priority shall be considered the nominated candidate.
4. If both parties send a candidate-used notification with candidates having the same priority, then the candidate chosen by the initiator shall be considered the nominated candidate (this is consistent with the rules in XEP-0166).

The parties shall use the nominated candidate for the data transfer. However, if the nominated candidate is of the ”proxy” type, then the peer has no way to know when it can send data. Therefore the party that offered the nominated candidate MUST do two things...
First, it activates the bytestream, as described in XEP-0065:

Listing 9: Responder activates the bytestream at proxy

```
<iq from='juliet@capulet.lit/balcony'
    id='vy1fa63k'
    to='streamer.shakespeare.lit'
    type='set'>
  <query xmlns='http://jabber.org/protocol/bytestreams'
         sid='a73sjjvkla37jfea'>
    <activate>romeo@montague.lit/orchard</activate>
  </query>
</iq>
```

Listing 10: Proxy informs responder of activation

```
<iq from='streamer.shakespeare.lit'
    id='vy1fa63k'
    to='juliet@capulet.lit/balcony'
    type='result'/>
```

Second, it sends an activated notification to the peer; it does so by sending a transport-info message containing an <activated/> element:

Listing 11: Responder informs initiator that bytestream has been activated

```
<iq from='juliet@capulet.lit/balcony'
    id='bv73bs91'
    to='romeo@montague.lit/orchard'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-info'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <transport xmlns='urn:xmpp:jingle:transports:s5b:1'
                 sid='vj3hs98y'>
        <activated cid='xmdh4b7i'/>
      </transport>
    </content>
  </jingle>
</iq>
```

If the nominated candidate is of the proxy type and either party cannot connect to the proxy (for example because of a restrictive firewall), the failing party shall send a transport-info

message containing an <proxy-error/> element.

Listing 12: Responder informs initiator of inability to connect to the proxy

```
<iq from='juliet@capulet.lit/balcony'
    id='bv73bs91'
    to='romeo@montague.lit/orchard'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-info'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <transport xmlns='urn:xmpp:jingle:transports:s5b:1'
                 sid='vj3hs98y'>
        <proxy-error/>
      </transport>
    </content>
  </jingle>
</iq>
```

The parties shall then consider the bytestream unsuccessful and SHOULD attempt to fall back to another transport as described in Fallback Methods.

2.5 Exchanging Data

Once the parties have chosen (and if necessary activated) a streamhost, they can exchange data as defined in XEP-0065.

2.6 Closing the Bytestream

Once the parties have finished using the bytestream (e.g., because a complete file has been sent), either party can send a Jingle session-terminate action.

Listing 13: Initiator terminates the session

```
<iq from='romeo@montague.lit/orchard'
    id='hz81vf48'
    to='juliet@capulet.lit/balcony'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='session-terminate'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <reason><success/></reason>
  </jingle>
```

```
</iq>
```

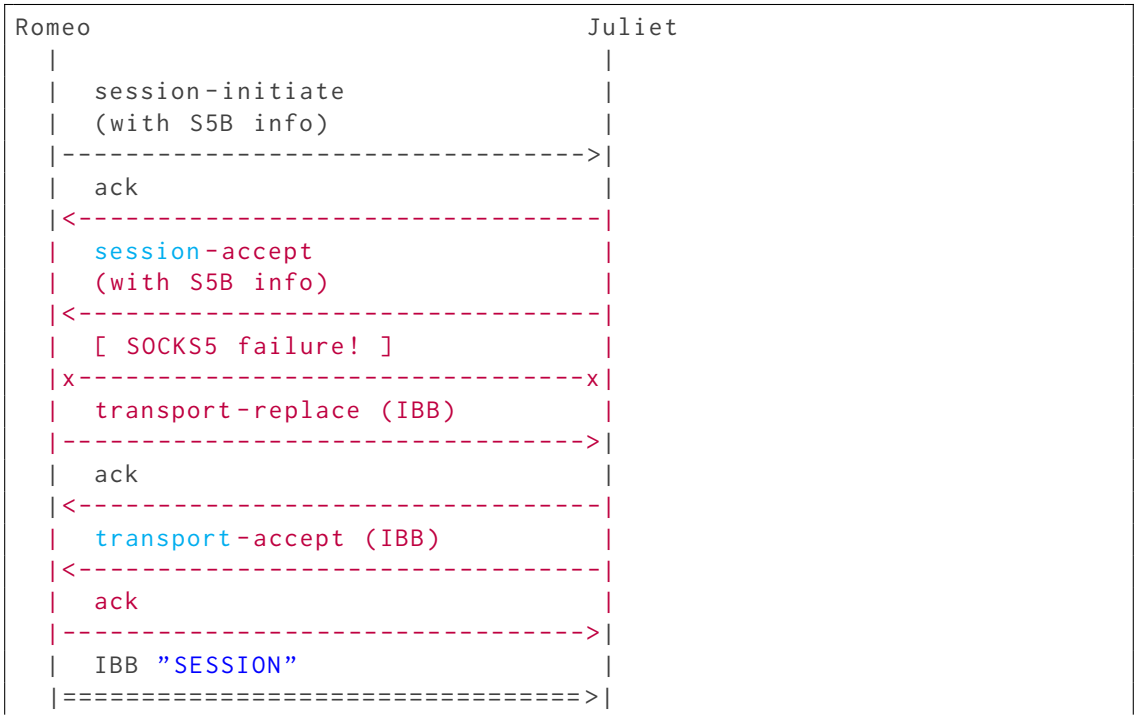
The other party then acknowledges the session-terminate and the Jingle session is finished.

Listing 14: Responder acknowledges session-terminate

```
<iq from='juliet@capulet.lit/balcony'
    id='hz81vf48'
    to='romeo@montague.lit/orchard'
    type='result'/>
```

3 Fallback Methods

If the SOCKS5 Bytestreams negotiation fails, the parties might want to "fall back" to another transport. Currently the transport of last resort for a streaming exchange is In-Band Bytestreams (XEP-0047) [6] as described for Jingle in Jingle In-Band Bytestreams Transport Method (XEP-0261) [7], however if other transport methods are defined in the future (e.g. RFC 6544 [8]) then clients could fall back to those methods instead of IBB. The protocol flow for fallback from S5B to IBB is as follows.



[6] XEP-0047: In-Band Bytestreams <https://xmpp.org/extensions/xep-0047.html>.
[7] XEP-0261: Jingle In-Band Bytestreams Transport Method <https://xmpp.org/extensions/xep-0261.html>.
[8] RFC 6544: TCP Candidates with Interactive Connectivity Establishment (ICE) <http://tools.ietf.org/html/rfc6544>.

```
|  terminate                        |
|<----------------------------------|
|  ack                              |
|---------------------------------->|
|                                   |
```

First the initiator sends a Jingle session-initiate, in this case with a transport of SOCKS5 Bytestreams. The protocol flow is exactly the same as described above. If both parties are unable to connect to a candidate provided by the peer, they send candidate-error messages to indicate that SOCKS5 has failed. The initiator MUST either terminate the Jingle session with a Jingle reason of <connectivity-error/> or replace the transport with something else using the transport-replace action. Typically the fallback option is IBB (see, for example, Jingle File Transfer (XEP-0234) [9]). Therefore the initiator sends a transport-replace action including a transport of IBB.

Listing 15: Initiator replaces transport with IBB

```
<iq from='romeo@montague.lit/orchard'
    id='hs92n57'
    to='juliet@capulet.lit/balcony'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-replace'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <transport xmlns='urn:xmpp:jingle:transports:ibb:1'
                 block-size='4096'
                 sid='ch3d9s71'/>
    </content>
  </jingle>
</iq>
```

The responder then acknowledges the transport-replace action.

Listing 16: Responder acknowledges transport-replace

```
<iq from='juliet@capulet.lit/balcony'
    id='hs92n57'
    to='romeo@montague.lit/orchard'
    type='result'/>
```

If the transport replacement is acceptable, the responder then sends a transport-accept action to the initiator (if not, the responder sends a transport-reject action). If the responder wishes to use a smaller block size than the one specified in the transport-replace offer, this can be done by specifying a block-size attribute in the transport-accept action.

[9] XEP-0234: Jingle File Transfer <https://xmpp.org/extensions/xep-0234.html>.

Listing 17: Responder sends transport-accept

```
<iq from='juliet@capulet.lit/balcony'
    id='is71ns63'
    to='romeo@montague.lit/orchard'
    type='set'>
  <jingle xmlns='urn:xmpp:jingle:1'
          action='transport-accept'
          initiator='romeo@montague.lit/orchard'
          sid='a73sjjvkla37jfea'>
    <content creator='initiator' name='ex'>
      <transport xmlns='urn:xmpp:jingle:transports:ibb:1'
                 block-size='2048'
                 sid='ch3d9s71'/>
    </content>
  </jingle>
</iq>
```

The initiator acknowledges the Jingle transport-accept action.

Listing 18: Initiator acknowledges transport-accept

```
<iq from='juliet@capulet.lit/balcony'
    id='is71ns63'
    to='romeo@montague.lit/orchard'
    type='result'/>
```

Now the parties can send data using In-Band Bytestreams as defined in XEP-0261 and XEP-0047.

4 Processing Rules and Usage Guidelines

The same processing rules and usage guidelines defined in XEP-0065 apply to the Jingle S5B Transport Method. This document adds the following implementation suggestions in the context of Jingle:

1. Try the offered candidates in the order of their priority, from highest to lowest.
2. Stagger the connection attempts (e.g., initiate communications with the highest-priority candidate, then wait 200ms before initiating communications with the second-highest-priority candidate).
3. To increase the potential for using a direct connection, consider waiting a bit longer than 200ms to initiate communications with proxy candidates.

5 Determining Support

To advertise its support for the Jingle SOCKS5 Bytestreams Transport Method, when replying to Service Discovery (XEP-0030) [10] information requests an entity MUST return URNs for any version of this protocol that the entity supports -- e.g., "urn:xmpp:jingle:transports:s5b:1" for this version (see Namespace Versioning regarding the possibility of incrementing the version number).

Listing 19: Service discovery information request

```
<iq from='romeo@montague.lit/orchard'
    id='uw72g176'
    to='juliet@capulet.lit/balcony'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 20: Service discovery information response

```
<iq from='juliet@capulet.lit/balcony'
    id='uw72g176'
    to='romeo@montague.lit/orchard'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <feature var='urn:xmpp:jingle:1'/>
    <feature var='urn:xmpp:jingle:transports:s5b:1'/>
  </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) [11]. However, if an application has not received entity capabilities information from an entity, it SHOULD use explicit service discovery instead.

6 Security Considerations

6.1 Sharing IP Addresses

The exchange of candidates might result 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 can consider that action to imply approval of IP address sharing.

[10] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.
[11] XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.

6.2 Encryption of Media

This specification, like XEP-0065 before it, does not directly support end-to-end encryption of the media sent over the transport.

7 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [12].

8 XMPP Registrar Considerations

8.1 Protocol Namespaces

The XMPP Registrar [13] includes 'urn:xmpp:jingle:transports:s5b:1' in its registry of protocol namespaces at <https://xmpp.org/registrar/namespaces.html>, as described in Section 4 of XMPP Registrar Function (XEP-0053) [14].

8.2 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.

8.3 Jingle Transport Methods

The XMPP Registrar [15] includes "jingle-s5b" in its registry of Jingle transport methods at <https://xmpp.org/registrar/jingle-transports.html>. The registry submission is as follows:

[12] 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/>.

[13] 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/>.

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

[15] 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/>.

```
<transport>
  <name>s5b</name>
  <desc>
    A method for negotiating data exchange over SOCKS5 Bytestreams.
  </desc>
  <type>streaming</type>
  <doc>XEP-0260</doc>
</transport>
```

9 Schema

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:jingle:transports:s5b:1'
    xmlns='urn:xmpp:jingle:transports:s5b:1'
    elementFormDefault='qualified'>

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0260: http://www.xmpp.org/extensions/xep-0260.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='transport'>
    <xs:complexType>
      <xs:choice>
        <xs:element ref='activated'
                    minOccurs='0' maxOccurs='1'/>
        <xs:element ref='candidate'
                    minOccurs='0' maxOccurs='unbounded'/>
        <xs:element name='candidate-error'
                    minOccurs='0' maxOccurs='1' type='empty'/>
        <xs:element ref='candidate-used'
                    minOccurs='0' maxOccurs='1'/>
        <xs:element name='proxy-error'
                    minOccurs='0' maxOccurs='1' type='empty'/>
      </xs:choice>
      <xs:attribute name='dstaddr' use='optional' type='xs:string'/>
      <xs:attribute name='mode' use='optional' default='tcp'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='tcp'/>
            <xs:enumeration value='udp'/>
          </xs:restriction>
```

```
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name='sid' type='xs:string' use='optional'/>
  </xs:complexType>
</xs:element>

<xs:element name='candidate'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='empty'>
        <xs:attribute name='cid'
                      type='xs:string' use='required'/>
        <xs:attribute name='host'
                      type='xs:string' use='required'/>
        <xs:attribute name='jid'
                      type='xs:string' use='required'/>
        <xs:attribute name='port'
                      type='xs:positiveInteger' use='optional'/>
        <xs:attribute name='priority'
                      type='xs:positiveInteger' use='required'/>
        <xs:attribute name='type'
                      use='optional' default='direct'>
          <xs:simpleType>
            <xs:restriction base='xs:NCName'>
              <xs:enumeration value='assisted'/>
              <xs:enumeration value='direct'/>
              <xs:enumeration value='proxy'/>
              <xs:enumeration value='tunnel'/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name='candidate-used'>
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base='empty'>
        <xs:attribute name='cid' type='xs:string' use='required'/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name='activated'>
  <xs:complexType>
    <xs:simpleContent>
```

```
        <xs:extension base='empty'>
          <xs:attribute name='cid' type='xs:string' use='required'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

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

</xs:schema>
```

10 Acknowledgements

Thanks to Steffen Larsen, Florian Schmaus, Kevin Smith, and Remko Tronçon for their feedback.