# XEP-0095: Stream Initiation


Thomas Muldowney
mailto:temas@jabber.org
xmpp:temas@jabber.org

Matthew Miller
mailto:linuxwolf@outer-planes.net
xmpp:linuxwolf@outer-planes.net

Ryan Eatmon
mailto:reatmon@jabber.org
xmpp:reatmon@jabber.org



2004-04-13
Version 1.1

| Status | Type | Short Name |
|---|---|---|
| Draft | Standards Track | si |



This specification defines an XMPP protocol extension for initiating a data stream between any two XMPP entities. The protocol includes the ability to include metadata about the stream and provides a pluggable framework so that various profiles of stream initiation can be defined for particular use cases (such as file transfer).

# Legal

## Copyright



## Permissions



## Warranty



## Liability



## Conformance

# Contents

# 1 Introduction

As the Jabber protocols are extended beyond basic messaging and presence, the need has arisen for a generic protocol that can be used to negotiate content streams between any two entities. Such streams might be in-band, but more likely will be out-of-band, binary streams used in applications such as file transfer, voice chat, and video conferencing. This document provides a method for negotiating such streams, including meta-information about the intended usage of the stream.

# 2 Requirements

- The defined protocol will allow for negotiation of a common stream.
- The defined protocol will allow for meta-information to be sent about the stream usage.
- The defined protocol will not be required for stream usage.

# 3 Use Case

The process for stream negotiation is as follows:

1. Sender discovers if Receiver implements the desired profile. [E1]
2. Sender offers a stream initiation. [E2]
3. Receiver accepts stream initiation.
4. Sender and receiver prepare for using negotiated profile and stream, EUC

Error Conditions:

1. The Receiver does not support the desired profile, EUC
2. Receiver rejects the stream initiation, EUC

## 3.1 Discovery

Before a Stream Initiation is attempted the Sender should be sure that the Receiver supports both Stream Initiation and the specific profile that they wish to use. This is typically accomplished using Service Discovery (XEP-0030) [1]:

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

Listing 1: Requesting Disco Information From Receiver

```
<iq type='get'
    to='receiver@jabber.org/resource'
    from='sender@jabber.org/resource'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

The Receiver advertises the "http://jabber.org/protocol/si" namespace as a feature to represent that they implement this document. The specific profiles are also announced this way; they can be found by looking for "http://jabber.org/protocol/si/profile/profile-name". Shown in the result is a potential file transfer profile:

Listing 2: Receiver Disco Information Result

```
<iq type='result'
    to='sender@jabber.org/resource'
    from='receiver@jabber.org/resource'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    ...
    <feature var='http://jabber.org/protocol/si'/>
    <feature var='http://jabber.org/protocol/si/profile/file-transfer'
       />
    ...
  </query>
</iq>
```

### 3.2 Negotiating Profile and Stream

Once support is determined, the Sender starts the negotiation with the Receiver by sending an <iq/> stanza of type "set", such as in the following example from SI File Transfer (XEP-0096) [2]:

Listing 3: Offer Stream Initiation

```
<iq type='set' id='offer1' to='receiver@jabber.org/resource'>
  <si xmlns='http://jabber.org/protocol/si'
      id='a0'
      mime-type='text/plain'
      profile='http://jabber.org/protocol/si/profile/file-transfer'>
    <file xmlns='http://jabber.org/protocol/si/profile/file-transfer'
          name='test.txt'
          size='1022'>
      <desc>This is info about the file.</desc>
    </file>
```

[2] XEP-0096: SI File Transfer <https://xmpp.org/extensions/xep-0096.html>.

```
    <feature xmlns='http://jabber.org/protocol/feature-neg'>
      <x xmlns='jabber:x:data' type='form'>
        <field var='stream-method' type='list-single'>
          <option><value>http://jabber.org/protocol/bytestreams</value
              ></option>
          <option><value>jabber:iq:oob</value></option>
          <option><value>http://jabber.org/protocol/ibb</value></
              option>
        </field>
      </x>
    </feature>
  </si>
</iq>
```

At this point the Receiver can view the profile and other information to decide if they wish to accept the Stream Initiation. If acceptable the Receiver MUST select one of the presented stream types to use.

Listing 4: Accept Stream Initiation

```
<iq type='result' to='sender@jabber.org/resource' id='offer1'>
  <si xmlns='http://jabber.org/protocol/si'>
    <feature xmlns='http://jabber.org/protocol/feature-neg'>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='stream-method'>
          <value>http://jabber.org/protocol/bytestreams</value>
        </field>
      </x>
    </feature>
  </si>
</iq>
```

If the profile describes data to be sent back in the result it MUST be present as described in the profile's specification.

Listing 5: Accept Stream Initiation with Profile

```
<iq type='result' to='sender@jabber.org/resource' id='offer1'>
  <si xmlns='http://jabber.org/protocol/si'>
    <file xmlns='http://jabber.org/protocol/si/profile/profile-name'>
      <value>returned value</value>
    </file>
    <feature xmlns='http://jabber.org/protocol/feature-neg'>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='stream-method'>
          <value>http://jabber.org/protocol/bytestreams</value>
        </field>
      </x>
    </feature>
```

```
  </si>
</iq>
```

If none of the stream types are acceptable, or if the profile is not understood, the Receiver MUST reply with a ”bad request” error:

Listing 6: No Valid Streams

```
<iq type='error' to='sender@jabber.org/resource' id='offer1'>
  <error code='400' type='cancel'>
    <bad-request xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <no-valid-streams xmlns='http://jabber.org/protocol/si'/>
  </error>
</iq>
```

Listing 7: Profile not understood

```
<iq type='error' to='sender@jabber.org/resource' id='offer1'>
  <error code='400' type='cancel'>
    <bad-request xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <bad-profile xmlns='http://jabber.org/protocol/si'/>
  </error>
</iq>
```

If the Receiver rejects the request, they reply with a ”forbidden” error:

Listing 8: Rejecting Stream Initiation

```
<iq type='error' to='sender@jabber.org/resource' id='offer1'>
  <error code='403' type='cancel'>
    <forbidden xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <text xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'>Offer Declined</
       text>
  </error>
</iq>
```

### 3.3 Preparing the Transfer

At this point, the Sender and Receiver make any preparations necessary for the stream to be used. The exact process is specific to each protocol, and is beyond the scope of this document. This step now marks the end of SI’s responsibilities.

# 4 Formal Definition

## 4.1 <si/> Root Element

The <si/> element is the root element for this protocol. It is an identifiable container for all the information necessary for negotiation and signalling. It contains attributes for the identifier, intended MIME-type, and profile. The contents convey stream-negotation and profile information.
The ”id” attribute is an opaque identifier. This attribute MUST be present on type='set', and MUST be a valid string. This SHOULD NOT be sent back on type='result', since the <iq/> ”id” attribute provides the only context needed. This value is generated by the Sender, and the same value MUST be used throughout a session when talking to the Receiver.
The ”mime-type” attribute identifies the MIME-type for the data across the stream. This attribute MUST be a valid MIME-type as registered with the Internet Assigned Numbers Authority (IANA) [3] (specifically, as listed at <http://www.iana.org/assignments/media-types>). During negotiation, this attribute SHOULD be present, and is otherwise not required. If not included during negotiation, its value is assumed to be ”application/octet-stream”.
The ”profile” attribute defines the SI profile in use. This value MUST be present during negotiation, and is the namespace of the profile to use.
When the Sender first negotiates a Stream Initiation, all of the attributes SHOULD be present, and the id” and ”profile” MUST be present. The contents MUST contain one profile, in the namespace declared in the ”profile” attribute, and the feature negotiation for the stream. The feature negotiation MUST contain at least one option and use the field var ”stream-method”.
When the Receiver accepts a Stream Initiation, the <si/> element SHOULD NOT possess any attributes. The selected stream MUST be in the feature negotiation for the stream. There MUST only be one selected stream.

## 4.2 Error Codes

To simplify the discussion on error conditions, this document uses the following mapping between namespace URIs and namespace prefixes[4].

| Prefix | URI |
|---|---|
| xmpp | urn:ietf:params:xml:ns:xmpp-stanzas |
| si | http://jabber.org/protocol/si |

[3] 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/>.
[4] This mapping is provided for the purpose of simplifying this discussion, and is not intended to be used in the actual protocol.

Below are the most common errors that can result.

| Error Code | Error Type | General Condition | Specific Condition | Description |
|---|---|---|---|---|
| 400 | cancel | <xmpp:bad-request/> | <si:no-valid-streams/> | None of the available streams are acceptable. |
| 400 | modify | <xmpp:bad-request/> | <si:bad-profile/> | The profile is not understood or invalid. The profile MAY supply a profile-specific error condition. |
| 403 | cancel | <xmpp:forbidden/> | NONE | The stream is rejected. |

For further information about the relationship between XMPP error handling and ”legacy” (HTTP-style) error codes, refer to Error Condition Mappings (XEP-0086) [5].

# 5 Implementation Notes

## 5.1 Profiles

Stream Initiation on its own is of limited use; the Receiver almost always requires some reason for SI. Knowing this allows the Receiver to make a more educated choice about whether or not to accept the stream. This information is provided in Stream Initiation via a *profile*. A profile is a collection of information that describes the reason for and structure of the stream data, including what the data represents and what stream protocols are expected to be supported. The initial request for Stream Initiation MUST have only one profile, and this profile is in its own namespace. The profile is indicated not only by the presence of a ”root” element in that particular namespace, but also be the ”profile” attribute in <si/> The SUGGESTED format for profile namespaces is:

```
http://jabber.org/protocol/si/profile/profile-name
```

The information that the profile presents SHOULD be defined in an official XEP. The XEP defining the profile SHOULD contain explanations of what the profile consists of and MUST define the profile in a complete manner using DTD, Schema or another appropiate definition language.

[5] XEP-0086: Error Condition Mappings <https://xmpp.org/extensions/xep-0086.html>.

A profile SHOULD define what stream protocols MUST be supported, and MUST define what stream protocols MAY be supported. If a profile specifies only a single stream protocol that MUST be supported (even if others MAY also be supported), the ”fneg” for the stream protocol may be omitted from the initial <si/>; the receiving entity then assumes the stream protocol that MUST be supported is the one to use.
This document does not define any profiles, nor does it place any restrictions on what type of information a profile should detail. Other specifications will define profiles to be used with Stream Initiation.

### 5.2 Stream Interaction

While Stream Initiation is not directly required for stream usage, it does provide many benefits. In order to fully appreciate these benefits, streams must link the Stream Initiation to the stream. The ”id” attribute transported on the <si/> element is intended to do this. Once a session is fully negotiated, the value of the <si/> ”id” attribute is used during the actual stream negotiation as the protocol’s stream identifier attribute.
To be compatible to this document, a stream protocol MUST define a stream identifier (typically ”sid”), which MUST have a unique string representation. The stream protocol MUST be able to use any string identifier chosen during Stream Initiation, as long as the sending party does not use the same identifier more than once.

## 6 Security Considerations

Data security concerns are left to the profiles to define. Wire security concerns are left to the stream definitions.

## 7 IANA Considerations

This document uses the MIME types as recorded by the IANA, but no direct interaction with the IANA is necessary.

# 8 XMPP Registrar Considerations

## 8.1 Protocol Namespaces

The XMPP Registrar [6] includes the 'http://jabber.org/protocol/si' namespace in its registry of protocol namespaces.

## 8.2 Registries

### 8.2.1 Profiles Registry

The XMPP Registrar shall maintain a registry of stream initiation profiles, located at <https://xmpp.org/registrar/si-profiles.html>. Any such profile defined in a XMPP Extension Protocol specification MUST be submitted to the XMPP Registrar; profiles defined in non-standard protocol extensions SHOULD be submitted to the XMPP Registrar.
In order to submit new values to this registry, the registrant shall define an XML fragment of the following form and either include it in the relevant XMPP Extension Protocol or send it to the email address <registrar@xmpp.org>:

```
<profile>
  <name>The profile name</name>
  <doc>The specification that defines the profile</doc>
  <desc>A natural-language description of the profile</desc>
</profile>
```

# 9 XML Schema

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://jabber.org/protocol/si'
    xmlns='http://jabber.org/protocol/si'
    elementFormDefault='qualified'>

  <xs:import
      namespace='http://jabber.org/protocol/feature-neg'
      schemaLocation='http://www.xmpp.org/schemas/feature-neg.xsd'/>

  <xs:annotation>
    <xs:documentation>
```

[6] 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/>.

```
      The protocol documented by this schema is defined in
      XEP-0095: http://www.xmpp.org/extensions/xep-0095.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='si'>
    <xs:annotation>
      <xs:documentation>
        This is the root content element. All other elements in
        this namespace are for communicating error information.
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence xmlns:fneg='http://jabber.org/protocol/feature-neg'
          >
        <xs:any namespace='##other' minOccurs='0'/>
        <xs:element ref='fneg:feature'/>
      </xs:sequence>
      <xs:attribute name='id' type='xs:string' use='optional'/>
      <xs:attribute name='mime-type' type='xs:string' use='optional'/>
      <xs:attribute name='profile' type='xs:string' use='optional'/>
    </xs:complexType>
  </xs:element>

  <xs:element name='bad-profile' type='empty'/>
  <xs:element name='no-valid-streams' type='empty'/>

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

</xs:schema>
```