XEP-0386: Bind 2

Kevin Smith
mailto:kevin.smith@isode.com
xmpp:kevin.smith@isode.com

Matthew Wild
mailto:mwild1@gmail.com
xmpp:me@matthewwild.co.uk

2025-12-12
Version 1.1.0

Status	Type	Short Name
Draft	Standards Track	bind2

This specification provides a single-request replacement for several activities an XMPP client needs to do at startup.

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2024 by the XMPP Standards Foundation (XSF).

Permissions

Permission is hereby granted, free of charge, to any person obtaining a copy of this specification (the ”Specification”), to make use of the Specification without restriction, including without limitation the rights to implement the Specification in a software program, deploy the Specification in a network service, and copy, modify, merge, publish, translate, distribute, sublicense, or sell copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the condition that the foregoing copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification. Unless separate permission is granted, modified works that are redistributed shall not contain misleading information regarding the authors, title, number, or publisher of the Specification, and shall not claim endorsement of the modified works by the authors, any organization or project to which the authors belong, or the XMPP Standards Foundation.

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 **1**

2 Requirements **1**

3 Use Cases **1**

3.1 Discovering support 1

3.2 Performing the bind 2

3.2.1 Resource identifier generation 4

4 Security Considerations **5**

5 IANA Considerations **5**

6 XMPP Registrar Considerations **5**

7 XML Schema **5**

8 Acknowledgements **7**

1 Introduction

Every session on XMPP generally has a unique routable identifier, known as a ”resource”. Many details and rules about resources in XMPP can be found in RFC 6120 [1]. This core RFC also describes how to ”bind” a resource identifier to a session. This is a key part of session establishment for practically all XMPP clients. This specification describes an alternative protocol for resource binding than the one described in RFC 6120, based on Extensible SASL Profile (XEP-0388) [2].
As XMPP has grown more feature-rich over time, more steps have been introduced that clients are likely to perform at startup, e.g. resource binding, archive synchronisation, enabling Carbons. Some of these introduce race conditions - e.g. if a client synchronises the archive before enabling Carbons, it can miss stanzas sent between these events, or if it enables Carbons before synchronising the archive it can receive duplicate messages. It may also cause duplicate messages by combining archive synchronisation and receipt of offline messages, or by receipt of messages addressed to the full JID between resource binding and archive synchronisation. Therefore, this document provides a mechanism for atomically performing these operations to avoid these race conditions. It also allows the server to provide information to a client that is generally useful about the state of the user’s account.

2 Requirements

- Reduce round-trips and race conditions by providing a single atomic session establishment operation, with support for common session features such as Message Carbons (XEP-0280) [3], Message Archive Management (XEP-0313) [4] and Stream Management (XEP-0198) [5]. It should also be extensible to additional protocols as needed.
- Integrate with SASL2 (XEP-0388) for further round-trip reduction and simpler session establishment.

3 Use Cases

3.1 Discovering support

If a server supports Bind 2, it MUST advertise this within the SASL2 <inline/> element in the stream features, with a feature named ’bind’ in the namespace ’urn:xmpp:bind:0’.
Clients do not advertise support for Bind 2.
Bind 2 supports inline negotiation of certain features specific to a session. The features supported by the server for such inline negotiation MUST be included in an <inline/> child

[1] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.
[2] XEP-0388: Extensible SASL Profile <https://xmpp.org/extensions/xep-0388.html>.
[3] XEP-0280: Message Carbons <https://xmpp.org/extensions/xep-0280.html>.
[4] XEP-0313: Message Archive Management <https://xmpp.org/extensions/xep-0313.html>.
[5] XEP-0198: Stream Management <https://xmpp.org/extensions/xep-0198.html>.

element within the bind feature element. Each feature is listed as a <feature/> child element with a 'var' attribute indicating the extension's defined service discovery feature name or namespace.

Listing 1: Server advertises support in stream features

```
<stream:features>
  <authentication xmlns='urn:xmpp:sasl:2'>
    <mechanism>SCRAM-SHA-1</mechanism>
    <mechanism>SCRAM-SHA-1-PLUS</mechanism>
    <inline>
      <bind xmlns='urn:xmpp:bind:0'>
        <inline>
          <feature var="urn:xmpp:carbons:2" />
          <feature var="urn:xmpp:csi:0" />
          <feature var="urn:xmpp:sm:3" />
        </inline>
      </bind>
      <sm xmlns='urn:xmpp:sm:3'/>
    </inline>
  </authentication>
</stream:features>
```

Bind 2 is never supported without SASL2, and so servers without support for SASL2 MUST NOT advertise the feature. Servers supporting SASL2 and Bind 2 may continue to offer legacy resource binding to clients.

3.2 Performing the bind

To request resource binding, the client MUST include a <bind/> element, qualified by the 'urn:xmpp:bind:0' namespace, in its SASL2 <authenticate/> request.
The <bind/> element MAY contain the following child elements:

- <tag/>: This element contains a short text string that typically identifies the software the user is using, mostly useful for diagnostic purposes for users, operators and developers. This tag may be visible to other entities on the XMPP network (see Security Considerations).

Additionally, the <bind/> element MAY contain one or more child elements in other namespaces, representing features that the client requests to be automatically enabled for its new session.

Listing 2: Client provides a resource bind request

```
<authenticate xmlns='urn:xmpp:sasl:2' mechanism='SCRAM-SHA-1'>
```

```
  <initial-response>[base64 encoded SASL data]</initial-response>
  <bind xmlns='urn:xmpp:bind:0'>
    <tag>AwesomeXMPP</tag>
  </bind>
</authenticate>
```

Listing 3: Client provides a resource bind request, additionally requesting some session features

```
<authenticate xmlns='urn:xmpp:sasl:2' mechanism='SCRAM-SHA-1'>
  <initial-response>[base64 encoded SASL data]</initial-response>
  <bind xmlns='urn:xmpp:bind:0'>
    <tag>AwesomeXMPP</tag>
    <enable xmlns='urn:xmpp:carbons:2'/>
    <enable xmlns='urn:xmpp:sm:3'/>
    <inactive xmlns='urn:xmpp:csi:0'/>
  </bind>
</authenticate>
```

If the client included a <bind/> element in its SASL2 <authenticate/> then the server MUST process the bind request after authentication is successful (including any necessary subsequent SASL2 tasks), but before sending the <success/> response. Following the usual rules of SASL2, the bind request MUST NOT be processed (i.e. it should be ignored) if the authentication is not successful.
Note: If the client included a <resume/> element in its SASL2 negotiation, that MUST be processed first by the server. If that resumption is successful, the server MUST skip resource binding (a resumed session already has a resource bound) and MUST entirely ignore the <bind/> request. If resumption of the previous stream fails, the server MUST include the XEP-0198 failure in the response, and then MUST proceed to process the bind request to establish a new session for the client.
Upon processing the bind request, the server MUST perform several operations, including:

- Clear the offline messages for this user, if any, without sending them (as they will be provided by MAM).
- Generate a resource identifier (per the rules below) and bind it to the current stream
- Get the archive id of the newest stanza in the user's MAM archive
- Enable any additional features requested by the client for this session

Upon successful binding of a resource, the server SHOULD terminate any earlier sessions from the same client (identified by the <user-agent> 'id' attribute in its SASL2 authentication request).

3.2.1 Resource identifier generation

If the client provided a <tag/> element in its bind request, the text content of that element SHOULD be included as-is in the final resource identifier, subject to the necessary validation for resource identifiers. This tag can help with client identification and debugging. The RECOMMENDED format is to include the client tag as a prefix of the server-generated identifier, separated by a single '/' character: [client tag]/[server generated identifier]. For example, AwesomeXMPP/rQ7Lwut0CcxW6.
Servers MAY choose to assign stable resource identifiers to clients, i.e. ensuring the same client will receive the same resource identifier for every bind request it makes. If a server or deployment provides resource identifier stability, the generated identifier SHOULD remain stable for every bind request with the same <tag/> and SASL2 <user-agent> id. The SASL2 <user-agent> id itself MUST NOT be exposed by the server in the generated resource identifier.
After processing the bind request as described above, the server MUST respond with the SASL2 <success/> element, including the client's full JID in the <authorization-identifier/> element (specified in Extensible SASL Profile (XEP-0388) [6]), and a <bound/> element qualified by the 'urn:xmpp:bind:0' namespace, as in the following example:

Listing 4: Server responds after processing the bind

```
<success xmlns='urn:xmpp:sasl:2'>
  <authorization-identifier>user@example.com/AwesomeXMPP.4232f4d4</
     authorization-identifier>
  <bound xmlns='urn:xmpp:bind:0'>
    <metadata xmlns='urn:xmpp:mam:2'>
      <start id='YWxwaGEg' timestamp='2008-08-22T21:09:04Z' />
      <end id='b21lZ2Eg' timestamp='2020-04-20T14:34:21Z' />
    </metadata>
  </bound>
</success>
```

The server SHOULD include a <metadata/> element as defined by XEP-0313, describing the state of the user's message archive at the precise time of resource binding. This helps the client determine what queries it may need to perform to synchronise messages.
Interactions with certain other extensions are hereby defined in this document:

- If the client included a XEP-0198 <enable/> element in the requested features, Stream Management should be enabled at this point. The <bound/> response MUST contain the resulting XEP-0198 <enabled/> (or <failed/>) element. For full business rules, syntax and examples, see XEP-0198.
- If the client included a XEP-0280 <enable/> element in the requested features, Message Carbons MUST be enabled at this point. As this operation MUST succeed, no response is necessary.

[6] XEP-0388: Extensible SASL Profile <https://xmpp.org/extensions/xep-0388.html>.

- If the client included a XEP-0352 <active/> or <inactive/> element, the session must begin in that state.

The <bound/> response MUST also contain any defined responses to other enabled features, if any, though details of these are beyond the scope of this specification.

4 Security Considerations

This specification mostly combines existing protocols together. Security considerations defined in those XEPs should be heeded as normal.
The additional facility provided here to provide information on the user's archive is provided post-authentication and is only providing the user's data to the user.
Implementations must adhere to the security considerations defined in XEP-0388 regarding the inclusion of SASL2 requests and inline feature negotiation in TLS 0-RTT ("early data") extensions. That is, they MUST NOT be sent or processed, except when appropriate mitigations are in place (which are beyond the scope of this document, but may be defined by others).
As it forms part of the resource identifier and therefore the full JID of the session, the 'tag' value provided by the client (if any), may be visible to other XMPP entities on the network that the client communicates with or that have access to the user's presence. The simple name of the client can provide value to users, operators and developers diagnosing issues, and it generally will not reveal more information than would be already available through service discovery. Unless they are operating in fully trusted environments, clients MUST NOT use identifiers that might reveal private information about a user or their system (such as hostnames).

5 IANA Considerations

None.

6 XMPP Registrar Considerations

The urn:xmpp:bind:0 namespace must be registered..

7 XML Schema

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

<xs:schema
```

```
  xmlns:xs='http://www.w3.org/2001/XMLSchema'
  xmlns:mam='urn:xmpp:mam:2'
  xmlns='urn:xmpp:bind:0'
  targetNamespace='urn:xmpp:bind:0'
  elementFormDefault='qualified'>

<xs:annotation>
  <xs:documentation>
    The protocol documented by this schema is defined in
    XEP-0386: https://xmpp.org/extensions/xep-0386.html
  </xs:documentation>
</xs:annotation>

<xs:import namespace='urn:xmpp:mam:2'
  schemaLocation='xep-0313.xsd'/>

<xs:element name='bind'>
  <xs:complexType>
    <xs:choice>
      <xs:element ref='inline'/>
      <xs:sequence>
        <xs:element ref='tag' minOccurs='0'/>
        <xs:any namespace='##other' minOccurs='0' maxOccurs='
           unbounded' processContents='lax'/>
      </xs:sequence>
    </xs:choice>
  </xs:complexType>
</xs:element>

<xs:element name='inline'>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref='feature' maxOccurs='unbounded'/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name='feature'>
  <xs:complexType>
    <xs:attribute name='var' type='xs:string'/>
  </xs:complexType>
</xs:element>

<xs:element name='tag' type='xs:string'/>

<xs:element name='bound'>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref='mam:metadata' minOccurs='0'/>
```

```
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

8 Acknowledgements

Thanks to Daniel Gultsch, Philipp Hörist, Thilo Molitor and Andrzej Wójcik for their valuable support with feedback, suggestions and implementations.