XEP-0215: External Service Discovery

Peter Saint-Andre
mailto:xsf@stpeter.im
xmpp:peter@jabber.org
http://stpeter.im/

Sean Egan
mailto:seanegan@google.com
xmpp:seanegan@google.com

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

Lance Stout
mailto:lance@andyet.com
xmpp:lance@lance.im

2015-10-20
Version 0.7

Status	Type	Short Name
Deferred	Standards Track	extdisco

This document specifies an XMPP protocol extension for discovering services external to the XMPP network.

Legal

Copyright

Permissions

Warranty

Liability

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

An XMPP client or other entity might need to discover services external to the XMPP network in order to complete certain XMPP-related use cases. One example is the discovery of STUN servers (see RFC 5389 [1]) and TURN relays (see RFC 5766 [2]) for the sake of negotiating media exchanges via the Jingle ICE-UDP Transport Method (XEP-0176) [3]. [4] An XMPP entity can already discover such external services in several ways, including:

1. The service is specified in the application's default settings.
2. The service is manually added into the application's configuration by a human user.
3. The service is discovered via non-XMPP service discovery protocols, such as:
   - DNS SRV records (RFC 2782 [5])
   - Service Location Protocol (SLP; RFC 2608 [6])
   - The Dynamic Delegation Discovery System (DDDS; RFC 3401 [7])
   - The NAPTR profile of DDDS (RFC 3403 [8])
   - The S-NAPTR profile of DDDS (RFC 3958 [9])
   - The U-NAPTR profile of DDDS (RFC 4848 [10])

Unfortunately, some of the foregoing methods are subject to human error and others are either not widely available or cannot be deployed in wide range of scenarios (e.g., when the administrators of an XMPP service do not have access to DNS SRV records). Therefore, this document defines a way for an XMPP server or discovery service to provide information about external services, which might include extended information such as temporary credentials for authentication at such services. This method SHOULD be used only as a fallback when the relevant service discovery technologies (DNS SRV, DDDS, SLP, S-NAPTR, U-NAPTR, etc.) are not available to the XMPP entities involved (typically a client and server). This method

[1] RFC 5389: Session Traversal Utilities for NAT (STUN) <http://tools.ietf.org/html/rfc5389>.
[2] RFC 5766: Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN) <http://tools.ietf.org/html/rfc5766>.
[3] XEP-0176: Jingle ICE-UDP Transport Method <https://xmpp.org/extensions/xep-0176.html>.
[4] The protocol specified herein is functionally equivalent to the protocol currently used in the Google Talk service for discovery of STUN servers, as documented at <http://code.google.com/apis/talk/jep_extensions/jingleinfo.html>, but has been broadened in scope to address additional use cases if desired.
[5] RFC 2782: A DNS RR for specifying the location of services (DNS SRV) <http://tools.ietf.org/html/rfc2782>.
[6] RFC 2608: Service Location Protocol, Version 2 <http://tools.ietf.org/html/rfc2608>.
[7] RFC 3401: Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS <http://tools.ietf.org/html/rfc3401>.
[8] RFC 3403: Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database <http://tools.ietf.org/html/rfc3403>.
[9] RFC 3958: Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS) <http://tools.ietf.org/html/rfc3958>.
[10] RFC 4848: Domain-Based Application Service Location Using URIs and the Dynamic Delegation Discovery Service (DDDS) <http://tools.ietf.org/html/rfc4848>.

does not use Service Discovery (XEP-0030) [11] since that technology is designed for discovery of XMPP entities, not entities outside an XMPP network.

2 Protocol

In order to learn about external services known to an XMPP server or discovery service, a requesting entity (typically a client) sends an IQ-get containing an empty <services/> element qualified by the 'urn:xmpp:extdisco:2' namespace (see Protocol Namespaces regarding issuance of one or more permanent namespaces), typically to its own server but perhaps alternatively to a dedicated discovery service.
The responding entity (XMPP server or discovery service) SHOULD return the list of external services it is aware of, but MAY instead return an appropriate error, such as <service-unavailable/> if the responding entity does not support this protocol or <forbidden/> if the requesting entity does not have permission to receive the list of external services. Each service is encapsulated via a <service/> element.
Note: The processes by which a responding entity discovers external services for "proxying" to XMPP entities are out of scope for this specification.
The <service/> element MAY be empty or MAY include extended information about the service as described in the Extended Information section of this document.
The attributes of the <service/> element are summarized in the following table.

Name	Definition	Inclusion
action	When sending a push update, the action value indicates if the service is being added or deleted from the set of known services (or simply being modified). The defined values are "add", "remove", and "modify", where "add" is the default.	OPTIONAL
expires	A timestamp indicating when the provided username and password credentials will expire. The format MUST adhere to the dateTime format specified in XMPP Date and Time Profiles (XEP-0082) XEP-0082: XMPP Date and Time Profiles <https://xmpp.org/extensions/xep-0082.html>. and MUST be expressed in UTC.	OPTIONAL
host	Either a fully qualified domain name (FQDN) or an IP address (IPv4 or IPv6).	REQUIRED

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

Name	Definition	Inclusion
name	A friendly (human-readable) name or label for the service.	OPTIONAL
password	A service- or server-generated password for use at the service. *	OPTIONAL
port	The communications port to be used at the host.	RECOMMENDED
restricted	A boolean value indicating that username and password credentials are required and will need to be requested if not already provided (see Requesting Credentials).	OPTIONAL
transport	The underlying transport protocol to be used when communicating with the service (typically either TCP or UDP).	RECOMMENDED
type	The service type as registered with the XMPP Registrar 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/>..	REQUIRED
username	A service- or server-generated username for use at the service. *	OPTIONAL

* Note: The processes by which an external service might generate (or an XMPP server might negotiate) the username and password are outside the scope of this specification. One possible approach is for the XMPP server to generate a short-term authentication credential based on a private key shared with the external service.

3 Use Cases

3.1 Requesting All Services

A requesting entity requests all services by sending a <services/> element to its server or a discovery service.

Listing 1: Entity Requests All External Services

```
<iq from='bard@shakespeare.lit/globe'
```

```
    id='ul2bc7y6'
    to='shakespeare.lit'
    type='get'>
  <services xmlns='urn:xmpp:extdisco:2'/>
</iq>
```

Listing 2: XMPP Server Returns List

```
<iq from='shakespeare.lit'
    id='ul2bc7y6'
    to='bard@shakespeare.lit/globe'
    type='result'>
  <services xmlns='urn:xmpp:extdisco:2'>
    <service host='stun.shakespeare.lit'
             port='9998'
             transport='udp'
             type='stun'/>
    <service host='relay.shakespeare.lit'
             password='jj929jkj5sadjfj93v3n'
             port='9999'
             transport='udp'
             type='turn'
             username='nb78932lkjlskjfdb7g8'/>
    <service host='192.0.2.1'
             port='8888'
             transport='udp'
             type='stun'/>
    <service host='192.0.2.1'
             port='8889'
             password='93jn3bakj9s832lrjbbz'
             transport='udp'
             type='turn'
             username='auu98sjl2wk3e9fjdsl7'/>
    <service host='ftp.shakespeare.lit'
             name='Shakespearean File Server'
             password='guest'
             port='20'
             transport='tcp'
             type='ftp'
             username='guest'/>
  </services>
</iq>
```

3.2 Requesting Selected Services

A requesting entity requests services of a particular type by sending a <services/> element including a 'type' attribute specifying the service type of interest.

Listing 3: Entity Requests Selected Services

```
<iq from='bard@shakespeare.lit/globe'
    id='yv2c19f7'
    to='shakespeare.lit'
    type='get'>
  <services xmlns='urn:xmpp:extdisco:2' type='turn'/>
</iq>
```

Listing 4: XMPP Server Returns List

```
<iq from='shakespeare.lit'
    id='yv2c19f7'
    to='bard@shakespeare.lit/globe'
    type='result'>
  <services xmlns='urn:xmpp:extdisco:2'
            type='turn'>
    <service host='turn.shakespeare.lit'
             password='jj929jkj5sadjfj93v3n'
             port='9999'
             transport='udp'
             type='turn'
             username='nb78932lkjlskjfdb7g8'/>
    <service host='192.0.2.1'
             port='8889'
             password='93jn3bakj9s832lrjbbz'
             transport='udp'
             type='turn'
             username='auu98sjl2wk3e9fjdsl7'/>
  </services>
</iq>
```

If a requesting entity requests services of a particular type, the responding service MAY as needed send an updated list of the relevant services by ”pushing” the list to a requesting entity that has previously requested the list. However, it MUST NOT push updates to the requesting entity unless it has presence information about the requesting entity (e.g., because the requesting entity is connected to the XMPP server or because the requesting entity has shared presence with a remote discovery service). A push is an IQ set to the requesting entity containing a <services/> payload with updated data about services matching the requested type (e.g., new services or updated credentials). Each <service/> element SHOULD contain an 'action' attribute indicating if the service is being added, deleted, or modified.

Listing 5: Services Push

```
<iq from='shakespeare.lit'
    id='lh3f1vc7'
    to='bard@shakespeare.lit/globe'
    type='set'>
  <services xmlns='urn:xmpp:extdisco:2'
```

```
            type='turn'>
    <service action='add'
             host='stun.shakespeare.lit'
             port='9999'
             transport='udp'
             type='turn'
             username='1nas9dlm3hzl89d0b9v'
             password='gh9023ljjdk109iajqn'>
    <service action='add'
             host='192.0.2.2'
             port='7778'
             transport='udp'
             type='turn'
             username='bnsv120afg48snsdozp'
             password='zxp023na98dsfahn1kk'/>
    <service action='delete'
             host='192.0.2.1'
             port='8889'
             type='turn'/>
  </services>
</iq>
```

Upon receiving a push, the requesting entity would then send an IQ-result to the responding service in accordance with XMPP Core [12].

3.3 Requesting Credentials

An entity might know about an external service via DNS or some other means, but still might need short-term credentials to use the service. The entity can request credentials by sending a special request to the server composed of a <credentials/> element qualified by the 'urn:xmpp:extdisco:2' namespace and contains a <service/> element which MUST include the 'host' and 'type' attributes to identify the desired service (the 'port' attribute MAY be provided if there are multiple services with the same host and type but different ports).

Listing 6: Entity Requests Credentials at a Service

```
<iq from='bard@shakespeare.lit/globe'
    id='xi2cax48'
    to='shakespeare.lit'
    type='get'>
  <credentials xmlns='urn:xmpp:extdisco:2'>
    <service host='turn.shakespeare.lit' type='turn' />
  </credentials>
</iq>
```

[12] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.

The server then returns credentials if possible.

Listing 7: Server Returns Credentials

```
<iq from='shakespeare.lit'
    id='xi2cax48'
    to='bard@shakespeare.lit/globe'
    type='get'>
  <credentials xmlns='urn:xmpp:extdisco:2'>
    <service host='turn.shakespeare.lit'
             type='turn'
             password='jj929jkj5sadjfj93v3n'
             username='nb78932lkjlskjfdb7g8'/>
  </credentials>
</iq>
```

For TURN, the server might construct time-limited credentials as described in A REST API for Access to TURN Services [13].
There MAY be multiple <service/> elements in the result if more than one service matched the requested service identity (e.g., the same host provides service on multiple ports).
If the server cannot obtain credentials at the service, it returns an appropriate stanza error, such as <item-not-found/>, <remote-server-not-found/>, <remote-server-timeout/>, or <not-authorized/>.

4 Extended Information

If a server or service needs to include extended information, it SHOULD do so by including each bit of information as the XML character data of the <value/> child of a distinct <field/> element, with the entire set of fields contained within an <x/> element of type "result" qualified by the 'jabber:x:data' namespace (see Data Forms (XEP-0004) [14]); this <x/> element SHOULD be a child of the <service/> element qualified by the 'urn:xmpp:extdisco:2' namespace (see Protocol Namespaces regarding issuance of one or more permanent namespaces). Thus the IQ result SHOULD be of the following form:

```
<iq type='result'>
  <services xmlns='urn:xmpp:extdisco:2'>
    <service>
      <x type='result' xmlns='jabber:x:data'>
        <field var='[var-name]' label='[optional]'>
          <value>[var-value]</value>
        </field>
```

[13] A REST API For Access To TURN Services <http://tools.ietf.org/html/draft-uberti-behave-turn-rest>. Work in progress.
[14] XEP-0004: Data Forms <https://xmpp.org/extensions/xep-0004.html>.

```
        [ ... ]
      </x>
    </service>
  </services>
</iq>
```

Note: A <field/> element MAY contain more than one <value/> child if appropriate.
If the data fields are to be used in the context of a protocol approved by the XMPP Standards Foundation, they SHOULD be registered in accordance with the rules defined in Field Standardization for Data Forms (XEP-0068) [15], resulting in the inclusion of a <field/> element whose 'var' attribute has a value of "FORM_TYPE" and whose 'type' attribute has a value of "hidden".
Note: Although Service Discovery Extensions (XEP-0128) [16] specifies that an XMPP entity MUST NOT supply extended information about associated children communicated via the 'http://jabber.org/protocol/disco#info' namespace, that rule does not apply to External Service Discovery since services external to the XMPP network cannot communicate via XMPP.

5 Determining Support

If an XMPP entity supports this protocol, it MUST report that fact by including a service discovery feature of "urn:xmpp:extdisco:2" (see Protocol Namespaces regarding issuance of one or more permanent namespaces) in response to a Service Discovery (XEP-0030) [17] information request:

Listing 8: Service Discovery Information Request

```
<iq from='romeo@montague.lit/orchard'
    id='ix61z3m9'
    to='montague.lit'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 9: Service Discovery Information Response

```
<iq from='montague.lit'
    id='ix61z3m9'
    to='romeo@montague.lit/orchard'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <feature var='urn:xmpp:extdisco:2'/>
```

[15] XEP-0068: Field Data Standardization for Data Forms <https://xmpp.org/extensions/xep-0068.html>.
[16] XEP-0128: Service Discovery Extensions <https://xmpp.org/extensions/xep-0128.html>.
[17] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

```
  </query>
</iq>
```

6 Internationalization Considerations

If the requesting entity includes an 'xml:lang' attribute with its request, the responding entity SHOULD include appropriately internationalized text as the value of the 'name' attribute. No other attributes are human-readable.

7 Security Considerations

Because the responding entity (XMPP server or discovery service) functions as a "proxy" from external services to the XMPP network, it could modify the information it receives before passing it on to the requesting entity.

8 XMPP Registrar Considerations

8.1 Protocol Namespaces

This specification defines the following XML namespace:

- urn:xmpp:extdisco:2

Upon advancement of this specification from a status of Experimental to a status of Draft, the XMPP Registrar [18] shall add the foregoing namespace to the registry located at <https://xmpp.org/registrar/namespaces.html>, as described in Section 4 of XMPP Registrar Function (XEP-0053) [19].

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.

[18] 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/>.

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

8.3 External Service Types Registry

The XMPP Registrar shall maintain a registry of external service types and their associated transport protocol(s). Such service types will probably be derived from the IANA Port Numbers Registry [20], defined DNS SRV record types, defined DDDS records for NAPTR, S-NAPTR, and U-NAPTR, and IANA Service Location Protocol, Version 2 (SLPv2) Templates [21].

8.3.1 Process

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>:

```
<service>
  <name>the XML character data of the service type</name>
  <desc>a natural-language description of the service type</desc>
  <doc>the document that best defines the service type</doc>
</service>
```

The registrant can register more than one service type at a time, each contained in a separate <service/> element.

8.3.2 Registration

```
<service>
  <name>stun</name>
  <desc>a server that provides Session Traversal Utilities for NAT (
      STUN)</desc>
  <doc>RFC 5389</doc>
</service>

<service>
  <name>turn</name>
  <desc>a server that provides Traversal Using Relays around NAT (TURN
      )</desc>
  <doc>RFC 5766</doc>
</service>
```

9 XML Schema

[20] IANA registry of port numbers <http://www.iana.org/assignments/port-numbers>.
[21] IANA registry of parameters related to the Service Location Protocol templates <http://www.iana.org/assignments/svrloc-templates.htm>.

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:extdisco:2'
    xmlns='urn:xmpp:extdisco:2'
    elementFormDefault='qualified'>

  <xs:import
      namespace='jabber:x:data'
      schemaLocation='http://www.xmpp.org/schemas/x-data.xsd'/>

  <xs:element name='services'>
    <xs:complexType>
      <xs:sequence minOccurs='0'>
        <xs:element ref='service'/>
      </xs:sequence>
      <xs:attribute name='type' type='xs:NCName' use='optional'/>
    </xs:complexType>
  </xs:element>

  <xs:element name='credentials'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='service' minOccurs='0' maxOccurs='unbounded'/
            >
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='service'>
    <xs:complexType>
      <xs:sequence xmlns:xdata='jabber:x:data'>
        <xs:element ref='xdata:x' minOccurs='0'/>
      </xs:sequence>
      <xs:attribute name='action' use='optional' default='add'>
        <xs:simpleType>
          <xs:restriction base='xs:NCName'>
            <xs:enumeration value='add'/>
            <xs:enumeration value='delete'/>
            <xs:enumeration value='modify'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name='expires' type='xs:dateTime' use='optional'/>
      <xs:attribute name='host' type='xs:string' use='required'/>
      <xs:attribute name='name' type='xs:string' use='optional'/>
      <xs:attribute name='password' type='xs:string' use='optional'/>
      <xs:attribute name='port' type='xs:string' use='required'/>
```

```
      <xs:attribute name='restricted' type='xs:boolean' use='optional'
         />
      <xs:attribute name='transport' type='xs:NCName' use='optional'/>
      <xs:attribute name='type' type='xs:NCName' use='required'/>
      <xs:attribute name='username' type='xs:string' use='optional'/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

10 Acknowledgements

Thanks to Philipp Hancke, Justin Karneges, Evgeniy Khramtsov, and Unnikrishnan Vikrama Panicker for their feedback.