XEP-0248: PubSub Collection Nodes

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

Ralph Meijer
mailto:ralphm@ik.nu
xmpp:ralphm@ik.nu

Brian Cully
mailto:bjc@kublai.com
xmpp:bjc@kublai.com

2010-09-28
Version 0.2

Status	Type	Short Name
Deferred	Standards Track	NOT_YET_ASSIGNED

This specification defines the nature and handling of collection nodes in the XMPP publish-subscribe extension.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

Publish-Subscribe (XEP-0060) [1] defines an XMPP protocol extension for generic publish-subscribe features. However, it only allows notifications from nodes to which an entity is directly subscribed. It is useful in some circumstances to describe a relationship between nodes so that a publish on one node may be delivered via another node. For instance, if an entity is interested in notifications from a set of nodes the entity would discover each node somehow and then subscribe to them. With collection nodes, the entity would subscribe only to the collection which links the desired nodes, simplifying the subscription process.
In addition to simplifying the subscriber's usage, collection nodes also allow the owner to describe almost any type of relationship between nodes. Using various access models on different nodes the owner can also create almost any desired authorization semantics on a set of leaf nodes.
Note: Any use cases not described herein are described in XEP-0060. Also, this document does not show error flows related to the generic publish-subscribe use cases referenced herein, since they are exhaustively defined in XEP-0060. The reader is referred to XEP-0060 for all relevant protocol details related to the XMPP publish-subscribe extension.

2 Scope

This documents addresses the common requirements regarding configuration, publishing, subscribing, and notification semantics of collection nodes.

3 Glossary

The following terms are used in this document to refer to collection node-specific features.
Note: some of these terms are specified in greater detail within the body of this document.

Collection Node A type of node that contains other nodes but no published items (c.f. Leaf Node).

Leaf Node A type of node that contains published items but no other nodes (c.f. Collection Node).

Node Graph The network of nodes emitting from a given node which contains all its descendants.

Root Node An anonymous collection node used as the de facto beginning of a service's node graph.

[1] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.

Subscription Depth How deep the collection node graph will be traversed when determining whether notifications will be sent. May be any integer, 0 or greater, or "all".

Subscription Type The type of notification, either "nodes", "items", or "all" which the subscriber is interested in.

4 Preliminaries

4.1 Collection Nodes

Collection nodes link nodes together to unify notifications from a set of collection or leaf nodes. An entity can subscribe to the collection and receive notifications of any associated leaf nodes.
A collection node can link with any other node in order to create a directed acyclic graph (DAG). Collection nodes MUST NOT be linked in such a way as to produce a cyclic graph (*i.e.*, they cannot link to nodes that eventually link back to the initial node).
Collection nodes only contain other nodes and MUST NOT contain published items (therefore a collection MUST NOT support the "publish" feature or related features such as "persistent-items").

5 Entity Use Cases

5.1 Discovering Support for Collection Nodes

An entity might wish to discover if a service implements collection nodes; in order to do so, it sends a service discovery information ("disco#info") query to the component's JID using Service Discovery (XEP-0030) [2]. If a service supports collection nodes it MUST return a "pubsub#collections" feature. In addition, if the service supports associating a node with more than one collection it MUST return a feature of "pubsub#multi-collections".

Listing 1: Entity requests features from a service

```
<iq type='get'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 2: Service responds with support for collections

```
<iq type='result'
    from='pubsub.shakespeare.lit'
```

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

```
    to='francisco@denmark.lit/barracks'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    ...
    <feature var='http://jabber.org/protocol/pubsub#collections'/>
    <feature var='http://jabber.org/protocol/pubsub#multi-collections'
        />
    ...
  </query>
</iq>
```

5.2 Discover Nodes

The service discovery items (”disco#items”) protocol enables an entity to query a service for a list of associated items, which, in the case of collection nodes would consist of the children associated with a given node.

5.2.1 Request

Listing 3: Entity requests child node discovery

```
<iq type='get'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='disco1'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='blogs'/>
</iq>
```

5.2.2 Success Case

Listing 4: Service responds with child nodes

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='disco1'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='blogs'>
    <item node='Romeoance'
          name='Letters␣to␣my␣Beloved'
          jid='pubsub.shakespeare.lit'/>
    <item node='Julliennui'
          name='A␣Rose␣by␣Another␣Name'
          jid='pubsub.shakespeare.lit'/>
  </query>
```

```
</iq>
```

5.3 Notifications

5.3.1 Generating Notifications for Collections

If a notification on a child node is created and then delivered via the collection then the notifications generated by the service MUST contain additional information. The 'node' attribute of the <item/> or <node/> element contained in the notification message MUST specify the node identifier of the node that generated the notification (not the collection) and the <message/> stanza MUST contain Stanza Headers and Internet Metadata (XEP-0131) [3] that specifies the node identifier of the collection.
Note: The delivery options (such as "pubsub#deliver_payloads") are determined by the publishing leaf node, not by the collection node. If the owner of a collection node sets delivery options for a collection node, the service SHOULD ignore those options and apply the options set for the leaf node that publishes an item.
Item notifications are notifications about the contents of a leaf node, and are generated by a publish, retract, or purge request.

Listing 5: Subscriber receives a publish notification from a collection

```
<message to='francisco@denmark.lit' from='pubsub.shakespeare.lit'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <items node='princely_musings'>
      <item id='ae890ac52d0df67ed7cfdf51b644e901'>
        ...
      </item>
    </items>
  </event>
  <headers xmlns='http://jabber.org/protocol/shim'>
    <header name='Collection'>blogs</header>
  </headers>
</message>
```

Node notifications are notifications about nodes themselves, and are generated by a create, delete, or configure request.

Listing 6: Subscriber receives a creation notification from a collection

```
<message to='francisco@denmark.lit' from='pubsub.shakespeare.lit'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <create node='princely_musings'/>
  </event>
  <headers xmlns='http://jabber.org/protocol/shim'>
    <header name='Collection'>blogs</header>
```

[3] XEP-0131: Stanza Headers and Internet Metadata <https://xmpp.org/extensions/xep-0131.html>.

```
  </headers>
</message>
```

5.3.2 Node Association and Dissociation

If a collection node is configured to send notification of node associations and disassociations, the service shall send an event that contains a <collection/> element whose 'node' attribute specifies the NodeID of the collection; this element in turn contains an <associate/> or <dissociate/> element whose 'node' attribute specifies the NodeID of node that has been associated with the collection.

Listing 7: Notification of node association

```
<message from='pubsub.shakespeare.lit'
         to='francisco@denmark.lit'
         id='newnode1'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <collection node='some-collection'>
      <associate node='new-node-id'>
    </collection>
  </event>
</message>
```

Listing 8: Notification of node dissociation

```
<message from='pubsub.shakespeare.lit'
         to='francisco@denmark.lit'
         id='newnode1'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <collection node='some-collection'>
      <dissociate node='old-node-id'>
    </collection>
  </event>
</message>
```

The notification event MAY also include the node meta-data, formatted using the Data Forms (XEP-0004) [4] protocol.

Listing 9: Notification of node association

```
<message from='pubsub.shakespeare.lit'
         to='francisco@denmark.lit'
         id='newnode2'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <collection node='some-collection'>
      <associate node='new-node-id'>
```

[4] XEP-0004: Data Forms <https://xmpp.org/extensions/xep-0004.html>.

```
        <x xmlns='jabber:x:data' type='result'>
          <field var='FORM_TYPE' type='hidden'>
            <value>http://jabber.org/protocol/pubsub#meta-data</value>
          </field>
          <field var='pubsub#creation_date'>
            <value>2003-07-29T22:56Z</value>
          </field>
          <field var='pubsub#creator'>
            <value>hamlet@denmark.lit</value>
          </field>
          <field var='pubsub#description'>
            <value>Atom feed for my blog.</value>
          </field>
          <field var='pubsub#language'>
            <value>en</value>
          </field>
          <field var='pubsub#contact'>
            <value>bard@shakespeare.lit</value>
          </field>
          <field var='pubsub#owner'>
            <value>hamlet@denmark.lit</value>
          </field>
          <field var='pubsub#title'>
            <value>Princely Musings (Atom).</value>
          </field>
          <field var='pubsub#type'
            ><value>http://www.w3.org/2005/Atom</value>
          </field>
        </x>
      </node>
    </collection>
  </event>
</message>
```

6 Subscriber Use Cases

6.1 Subscribe to a Collection Node

A service that implements collection nodes SHOULD allow entities to subscribe to collection nodes (subject to access models and local security policies).
In addition to the subscription configuration options already defined in XEP-0060, there are two subscription configuration options specific to collection nodes:

- **pubsub#subscription_type**
  This subscription option enables the subscriber to subscribe either to notifications about items or notifications about nodes.

If the subscription type is ”items”, the subscriber shall be notified whenever any node contained in the collection has an item published to it, retracted from it, or the node is purged, as modified by the value of the ”pubsub#subscription_depth” option.
If the subscription type is ”nodes”, the subscriber shall be notified whenever a new node is added to the collection, removed from the collection, or the configuration of a node within the collection has changed, as modified by the value of the ”pubsub#subscription_depth” option.
If the subscription type is ”all”, the subscriber shall be notified about both ”items” and ”nodes” types of events, as modified by the value of the ”pubsub#subscription_depth” option.
The default value of this subscription option SHOULD be ”nodes”.

- **pubsub#subscription_depth**
  This subscription option enables the subscriber to specify how far to traverse the node graph when determining whether a notification will be sent. It may be any integer value, 0 or greater, or the value ”all” which means that any node within the collection will generate a notification.
  The default value of this subscription option SHOULD be ”1”.

In order to subscribe to a collection node, an entity MUST send a subscription request to the node; the subscription request MAY include subscription options, but this is not strictly necessary (especially if the entity does not wish to override the default settings for the ”pubsub#subscription_type” and ”pubsub#subscription_depth” options).

6.1.1 Request

Listing 10: Entity subscribes to a collection node (no configuration)

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='collsub1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscribe jid='francisco@denmark.lit'
               node='blogs'/>
  </pubsub>
</iq>
```

The subscriber will now receive notification of new first-level nodes created within the ”blogs” collection.

Listing 11: Entity subscribes to a collection node (with configuration)

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='collsub1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscribe jid='francisco@denmark.lit'
               node='blogs'/>
    <options>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#subscribe_options</
              value>
        </field>
        <field var='pubsub#subscription_type'>
          <value>items</value>
        </field>
        <field var='pubsub#subscription_depth'>
          <value>all</value>
        </field>
      </x>
    </options>
  </pubsub>
</iq>
```

6.1.2 Success Case

If the service allows the subscription it MUST inform the requesting entity that it is now subscribed.

Listing 12: Service responds with success

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='collsub1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscription node='blogs'
                  jid='francisco@denmark.lit'
                  subscription='subscribed'/>
  </pubsub>
</iq>
```

6.1.3 Error Cases

A service MAY allow an entity to subscribe to a collection node in two ways, once with a subscription of type ”nodes” (to receive notification of any new nodes added to the collection or the entire tree) and once with a subscription of type ”items” (to receive all items published

within the tree). However, a service SHOULD NOT allow an entity to subscribe twice to a collection node (once with a subscription depth of "1" and once with a subscription depth of "all") for the same subscription type, since two such subscriptions are unnecessary (a depth of "all" includes by definition a depth of "1"); in this case the service SHOULD return a <conflict/> error to the requesting entity.

Listing 13: Service does not allow mulitple subscriptions to the same node

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='collsub1'>
  <error type='cancel'>
    <conflict xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

Depending on the nature of the node graph, a subscription type of "items" and depth of "all" may result in an extremely large number of notifications. Therefore, a service MAY disallow such a combination of subscription options, in which case it MUST return a <not-allowed/> error to the requesting entity.

Listing 14: Service does not allow requested options

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='collsub1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

6.2 Retrieving Items on Collection Nodes

When an entity requests items on a collection node the service SHOULD return the items on any leaf nodes associated with it subject to the access model of the collection node.

6.2.1 Request

Listing 15: Subscriber requests all items on a collection

```
<iq type='get'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
```

```
    id='items1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <items node='blogs'/>
  </pubsub>
</iq>
```

6.2.2 Success Case

When a collection contains multiple nodes with items it MUST return multiple <items/> elements, one per node.

Listing 16: Service returns items on leaf nodes

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='items1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <items node='Romeoance'>
      <item id='368866411b877c30064a5f62b917cffe'>
        ...
      </item>
    </items>
    <items node='Julliennui'>
      <item id='3300659945416e274474e469a1f0154c'>
        ...
      </item>
    </items>
  </pubsub>
</iq>
```

6.2.3 Error Cases

Depending on the nature of the node graph it may be expensive to allow item retrieval from a collection node. Therefore the service MAY disallow item retrieval via collection nodes, in which case it MUST return a <feature-not-implemented/> error to the requesting entity.

Listing 17: Service cannot fulfil request

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='items1'>
  <error type='cancel'>
    <feature-not-implemented xmlns='urn:ietf:params:xml:ns:xmpp-
        stanzas'/>
```

```
  </error>
</iq>
```

7 Owner Use Cases

7.1 Create a New Collection Node

To create a new collection node, the requesting entity MUST include a Data Form containing a "pubsub#node_type" field whose <value/> element contains "collection". The default value for "pubsub#node_type" SHOULD be "leaf".

7.1.1 Request

Listing 18: Entity requests a new collection node

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='create3'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <create node='announcements'/>
    <configure>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#node_config</value>
        </field>
        <field var='pubsub#node_type'><value>collection</value></field
          >
      </x>
    </configure>
  </pubsub>
</iq>
```

7.1.2 Success Case

Listing 19: Service responds with success

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='create3'/>
```

7.1.3 Error Cases

In addition to the errors already defined for leaf node creation, there are several reasons why the collection node creation request might fail:

1. The service does not support collection nodes.
2. The service does not support creation of collection nodes.
3. The requesting entity does not have sufficient privileges to create collection nodes.

These error cases are described more fully in the following sections.
If the service does not support collection nodes, it MUST respond with a <feature-not-implemented/> error, specifying a pubsub-specific error condition of <unsupported/> and a feature of "collections".

Listing 20: Service does not support collection nodes

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='create3'>
  <error type='cancel'>
    <feature-not-implemented xmlns='urn:ietf:params:xml:ns:xmpp-
        stanzas'/>
    <unsupported xmlns='http://jabber.org/protocol/pubsub#errors'
                 feature='collections'/>
  </error>
</iq>
```

If the service supports collection nodes but does not allow new collection nodes to be created, it MUST respond with a <not-allowed/> error.

Listing 21: Service does not allow creation of collection nodes

```
<iq type='error'
    from='hamlet@denmark.lit/elsinore'
    to='pubsub.shakespeare.lit'
    id='create3'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the requesting entity has insufficient privileges to create new collections, the service MUST respond with a <forbidden/> error.

Listing 22: Requesting entity has insufficient privileges to create collection nodes

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='create3'>
  <error type='auth'>
    <forbidden xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

7.2 Configuring a Collection Node

7.2.1 Request

In addition to the node configuration options specified in XEP-0060, there are three additional node configuration options that a service which supports collection nodes MUST supply.

- **pubsub#node_type** Whether this is a "leaf" or "collection" node.
- **pubsub#collection** The parents of this node.
- **pubsub#children** The children of this node.

To associate the root node to the collection the <value/> element MUST be empty.
A service MAY offer some node configuration options that are specific to collection nodes and SHOULD NOT be provided in configuration forms related to leaf nodes. The following are RECOMMENDED:

- **pubsub#children_association_policy** The policy regarding who may associate child nodes with the collection (values: all, owner, whitelist).
- **pubsub#children_association_whitelist** The whitelist of entities that may associate child nodes with the collection.
- **pubsub#children_max** The maximum number of child nodes that may be associated with a collection.

Listing 23: Entity configures a collection node

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='config1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#owner'>
    <configure node='blogs'>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#node_config</value>
        </field>
        <field var='pubsub#node_type'>
          <value>collection</value>
        </field>
        <field var='pubsub#children'>
          <value>Romeoance</value>
          <value>Julliennui</value>
        </field>
        <field var='pubsub#collection'>
          <value/>
        </field>
      </x>
    </configure>
  </pubsub>
</iq>
```

7.2.2 Success Case

Listing 24: Service successfully updates configuration

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='config1'/>
```

7.2.3 Error Cases

Leaf nodes only contain published items and MUST NOT have any children. If an entity attempts to add children to a leaf node (either via "pubsub#children" on the leaf node or "pubsub#collection" on another node) the service MUST return a <not-allowed/> error with a pubsub-specific error condition of <invalid-options/>.

Listing 25: Attempt to add a leaf node as the parent of another node

```
<iq type='error'
    from='pubsub.shakespeare.lit'
```

```
    to='francisco@denmark.lit/barracks'
    id='config1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <invalid-options xmlns='http://jabber.org/protocol/pubsub#errors'/
       >
  </error>
</iq>
```

If the requesting entity is not authorized to add the node to a collection then the service MUST return a <forbidden/> error.

Listing 26: Entity is not authorized to add node to a collection

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='config1'>
  <error type='cancel'>
    <forbidden xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the configuration would exceed the maximum number of children allowed on a node, either because the node's "pubsub#children" exceeds it's own "pubsub#children_max" value or because adding this node to a parent via "pubsub#collection" would exceed the parent's "pubsub#children_max" value, the service MUST return a <not-allowed/> error with a pubsub-specific error condition of <max-nodes-exceeded/>.

Listing 27: Node would contain too many children

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='config1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <max-nodes-exceeded xmlns='http://jabber.org/protocol/pubsub#
       errors'/>
  </error>
</iq>
```

The service MUST NOT allow the node type to be changed. If it is attempted the service MUST return a <not-allowed/> error, specifying a pubsub-specific error condition of <invalid-options/>

Listing 28: Attempt to change node type

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='config1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <invalid-options xmlns='http://jabber.org/protocol/pubsub#errors'/
        >
  </error>
</iq>
```

The service MUST NOT allow a cycle to be created in the node graph (*e.g.*, node A to B to C to A). If an entity attempts to submit a configuration that would create a cycle the service MUST return a <not-allowed/> error, specifying a pubsub-specific error condition of <invalid-options/>.

Listing 29: Cycle created in node graph

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='config1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <invalid-options xmlns='http://jabber.org/protocol/pubsub#errors'/
        >
  </error>
</iq>
```

7.3 Deleting a Collection Node

7.3.1 Request

If a service supports collection node creation it MUST support collection node deletion.

Listing 30: Owner attempts to delete a collection node

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='delete1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#owner'>
    <delete node='blogs'/>
  </pubsub>
</iq>
```

7.3.2 Success Case

If no error occurs, the service MUST inform the owner of success.

Listing 31: Collection node was deleted

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='delete1'/>
```

7.3.3 Error Cases

If the requesting entity attempts to delete the root node, the service MUST return a <not-allowed/> error.

Listing 32: Node is the root

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='delete1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

7.4 Associating a Node to a Collection

7.4.1 Request

A service MAY allow collection nodes to have children associated with them without changing the rest of the configuration. If the service allows this an entity can send and <associate/> element with a 'node' attribute that contains the child node within a <collection/> element that posesses a 'node' attribute containing the parent node to the service.

Listing 33: Entity requests node association

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='assoc1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#owner'>
    <collection node='some-collection'>
      <associate node='new-child-node'/>
    </collection>
```

```
  </pubsub>
</iq>
```

7.4.2 Success Case

If the service allows the node association then it MUST confirm the association with an empty result.

Listing 34: Service associates the node

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit'
    id='assoc1'/>
```

7.4.3 Error Cases

Listing 35: Entity is not authorized to associate the node

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='assoc1'>
  <error type='cancel'>
    <forbidden xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the configuration would exceed the maximum number of children allowed on a node, either because the node's "pubsub#children" exceeds it's own "pubsub#children_max" value or because adding this node to a parent via "pubsub#collection" would exceed the parent's "pubsub#children_max" value, the service MUST return a <not-allowed/> error with a pubsub-specific error condition of <max-nodes-exceeded/>.

Listing 36: Node would contain too many children

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='assoc1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <max-nodes-exceeded xmlns='http://jabber.org/protocol/pubsub#
        errors'/>
  </error>
</iq>
```

The service MUST NOT allow a cycle to be created in the node graph (*e.g.*, node A to B to C to A). If an entity attempts to submit a configuration that would create a cycle the service MUST return a <not-allowed/> error, specifying a pubsub-specific error condition of <invalid-options/>.

Listing 37: Cycle created in node graph

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='assoc1'>
  <error type='cancel'>
    <not-allowed xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <invalid-options xmlns='http://jabber.org/protocol/pubsub#errors'/
        >
  </error>
</iq>
```

7.5 Dissociating a Node from a Collection

7.5.1 Request

A service MAY allow collection nodes to have children dissociated from them without changing the rest of the configuration. If the service allows this an entity can send and <dissociate/> element with a 'node' attribute that contains the child node within a <collection/> element that posesses a 'node' attribute containing the parent node to the service.

Listing 38: Entity requests node dissociation

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='dissoc1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#owner'>
    <collection node='some-collection'>
      <dissociate node='old-child-node'/>
    </collection>
  </pubsub>
</iq>
```

7.5.2 Success Case

If the service allows the node dissociation then it MUST confirm the association with an empty result.

Listing 39: Service dissociates the node

```
<iq type='result'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit'
    id='dissoc1'/>
```

7.5.3 Error Cases

If a dissociation is requested between two nodes that are not already associated then the service MUST return a <bad-request/> error.

Listing 40: Node is not associated

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit'
    id='dissoc1'>
  <error type='modify'>
    <bad-request xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

Listing 41: Entity is not authorized to dissociate the node

```
<iq type='error'
    from='pubsub.shakespeare.lit'
    to='francisco@denmark.lit/barracks'
    id='dissoc1'>
  <error type='cancel'>
    <forbidden xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

8 Implementation Notes

8.1 Root Node

To provide a starting point for service discovery a service SHOULD support a root node. A root node represents the node belonging to a given service and MUST be identified by the lack of a node identifier (*i.e.*, the address of the pubsub service itself, such as "pubsub.shakespeare.lit"). Because the root node is owned by the service itself an entity SHOULD NOT be allowed create, delete, or configure the root node.
If a node is created or configured without any parents specified, a service MAY automatically associate otherwise orphaned nodes directly to the root node. If a service automatically associates a node with the root it MUST reflect that in the node configuration data form.

Listing 42: Entity subscribes to the root node

```
<iq type='set'
    from='francisco@denmark.lit/barracks'
    to='pubsub.shakespeare.lit'
    id='root1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscribe jid='francisco@denmark.lit'/>
  </pubsub>
</iq>
```

8.2 Handling Collection Node Deletion

Deletion of collection nodes can have a number of side effects due to implementation. Depending on the nature of the collection any of the following MAY happen when a collection node is deleted:

- When the collection node is deleted and the child nodes have no other parents the child nodes are orphaned; meaning that they will have no parent node, but continue to exist.
- When the collection node is deleted and the child nodes have no other parents the child nodes are re-assigned as children of the root node.
- When the collection node is deleted and the child nodes have no other parents the child nodes are also deleted.
- When the collection node is deleted and the child nodes have at least one other node the child nodes MUST remain associated with remaining parent nodes.

8.3 Updating Node Configuration When Associating or Dissociating Nodes

Node configuration MUST always reflect the current state of the node graph. Because node configuration contains both a pointer to its parents as well as its children an update to a primary node's "pubsub#collection" value will change the value of the secondary node's "pubsub#children" value, and vice-versa. A service MAY send a notification of the configuration change on the secondary node to subscribers if "pubsub#notify_config" is enabled on the secondary node.

9 Security Considerations

9.1 Access Models

Collection nodes can be used to associate almost any node within the service, but only the access model of the collection node itself is used to determine what an entity is allowed to see. Therefore care should be taken that nodes are not linked in such a way as to leak private

data (e.g., from a ”closed” leaf node through an ”open” collection) unless that behavior is specifically desired.

10 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [5].

11 XMPP Registrar Considerations

Note: These options are in addition to the standard options described in XEP-0060 and related XEPs.

11.1 Service Discovery Features

```
<var>
  <name>pubsub#collections</name>
  <desc>Support for collection nodes</desc>
  <doc>XEP-0248</doc>
</var>
<var>
  <name>pubsub#multi-collections</name>
  <desc>Support for multiple collections on a node</desc>
  <doc>XEP-0248</doc>
</var>
```

11.2 Field Standardization

```
<form_type>
  <name>http://jabber.org/protocol/pubsub#subscribe_options</name>
  <doc>XEP-0248</doc>
  <desc>Options for collection node subscription</desc>
  <field
      var='pubsub#subscription_depth'
      type='text-single'>
      label='How far to traverse the node graph for notifications'/>
  <field
      var='pubsub#subscription_type'
```

[5] 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/>.

```
      type='list-single'>
    <option label='Receive␣notification␣of␣items␣only'>
      items
    </option>
    <option label='Receive␣notification␣of␣nodes␣only'>
      nodes
    </option>
    <option label='Receive␣notification␣of␣items␣and␣nodes'>
      all
    </option>
  </field>
</form_type>
<form_type>
  <name>http://jabber.org/protocol/pubsub#node_config</name>
  <doc>XEP-0248</doc>
  <desc>Options for collection node configuration</desc>
  <field
      var='pubsub#node_type'
      type='list-single'>
    <option label='The␣node␣contains␣items'>
      leaf
    </option>
    <option label='The␣node␣contains␣other␣nodes'>
      collection
    </option>
  </field>
  <field
      var='pubsub#collection'
      type='text-multi'
      label='The␣collections␣of␣which␣this␣node␣is␣a␣child'/>
  <field
      var='pubsub#children'
      type='text-multi'
      label='The␣nodes␣of␣which␣this␣node␣is␣a␣parent'/>
  <field
      var='pubsub#children_association_policy'
      type='list-single'>
    <option label='Only␣the␣owners␣of␣this␣node␣may␣associate␣other␣
       nodes␣to␣this␣collection'>
      owners
    </option>
    <option label='Only␣those␣on␣the␣children␣association␣whitelist␣
       may␣associate␣other␣nodes␣to␣this␣collection'>
      whitelist
    </option>
    <option label='Anyone␣may␣associate␣nodes␣with␣this␣collection'>
      all
    </option>
  </field>
```

```
  <field
      var='pubsub#children_association_whitelist'
      type='jid-multi'
      label='JIDs who can associate nodes to this collection'/>
  <field
      var='pubsub#children_max'
      type='text-single'
      label='The maximum number of children for this collection'/>
</form_type>
```

11.3 SHIM Headers

```
<header>
  <name>Collection</name>
  <desc>The node of subscription that sent a notification</desc>
  <doc>XEP-0248</doc>
</header>
```

12 XML Schema

REQUIRED.

13 Acknowledgements

Many thanks to Dave Cridland for his feedback and advice.