XML Object Query

midPoint search filters are inspired by LDAP search filters. The search filters are composed from simple clauses and are represented in XML, using native XML element hierarchy to represent clause nesting.

Examples

Searching for account by DN

```
<q:equal>
    <q:path>c:attributes/icfs:name</q:path>
    <q:value>cn=foobar,uo=people,dc=nlight,dc=eu</q:value>
</q:equal>
```

Searching for user by user name or full name (short polyString version)

```
<q:or>
    <q:equal>
        <q:path>c:name</q:path>
        <q:value>jack</q:value>
    </q:equal>
    <q:equal>
        <q:path>c:fullName</q:path>
        <q:value>cpt. Jack Sparrow</q:value>
    </q:equal>
</q:or>
```

Searching for user by user name or full name (long polyString version)

```
<q:or>
    <q:equal>
        <q:path>c:name</q:path>
        <q:value>
            <t:orig>jack</t:orig>
            <t:norm>jack</t:norm>
        </q:value>
    </q:equal>
    <q:equal>
        <q:path>c:fullName</q:path>
        <q:value>
            <t:orig>cpt. Jack Sparrow</t:orig>
            <t:norm>cpt jack sparrow</t:norm>
        </q:value>
    </q:equal>
</q:or>
```

Searching for account by resource and intent (account type)

```
<q:and>
    <q:ref>
        <q:path>c:resourceRef</q:path>
                <q:value>
                <q:oid>ef2bc900-76e0-59e2-86d6-004f02d30000</q:oid>
                </q:value>
    </q:ref>
    <q:equal>
        <q:path>c:intent</q:path>
        <q:value>default</q:value>
    </q:equal>
</q:and>
```

Searching for user by employee type and organization structure membership

```
<q:and>
    <q:equal>
        <q:path>c:employeeType</q:path>
        <q:value>FTE</q:value>
    </q:equal>
    <q:org>
        <q:ref>
            <q:oid>ef2bc900-76e0-59e2-86d6-004f02d30000</q:oid>
        </q:ref>
        <q:maxDepth>unbounded</q:maxDepth>
    </q:org>
</q:and>
```

The search filter language definition is part of Query Schema. See the `FilterType` and derived types in the XSD file for more details.

Filters with Expressions

Equals filter with expression

```
<q:equal>
    <q:path>name</q:path>
    <c:expression>
       <c:path>$c:account/c:attributes/ri:uid</c:path>
    </c:expression>
</q:equal>
```

Reference with inner expression

```
<q:org>
    <q:ref>
        <q:oid>
            <c:expression>
                <c:path>$role:orgRef</c:path>
            </c:expression>
        </q:oid>
    </q:ref>
    <q:maxDepth>unbounded</q:maxDepth>
</q:org>
```

Reference with inner search filter (NOT YET IMPLEMENTED)

```
<q:org>
    <q:ref>
        <q:query>
                <q:filter>
                        <q:equals>
                                <q:path>name</q:path>
                                <c:expression>
                                        <c:path>$role/extension/orgName</c:path>
                                </c:expression>
                        </q:equals>
                </q:filter>
        <q:query>
    </q:ref>
    <q:maxDepth>unbounded</q:maxDepth>
</q:org>
```

Note the namespaces.

Motivation

We need a query language for midPoint that can be easily implemented on top of all the data stores that we plan to use. The least powerful search from the set of expected data stores is most likely LDAP. The good news is that we do not really need anything more complex than LDAP search filters, maybe "standardizing" some of the LDAP features that are usually LDAP extensions or special attributes (such as objectClass or VLV).

We want this to be pure XML, so it can go well in WSDL interface definitions, can be checked by XSD schema and so on. We also want them to be easily readable, so we can get visibility to the system.

SPMLv2 search clauses are very generic and too complex for our purposes. XQuery is not XML and has more flexibility that we need (and that means is much more complex). We do not need full XPath power, as we have only a flat set of properties and do not support XML structures of unconstrained depth.

External links

- What is midPoint Open Source Identity & Access Management
- Evolveum - Team of IAM professionals who developed midPoint