Paged search on resources HOWTO - UPDATED

There is a new feature of the ConnId framework since version 1.4: **paged search**. The client can specify that it would like to retrieve not all query results in one operation, but only a part of it - specified by an offset and an item count.

By default, midPoint does not utilize the paged search when retrieving accounts from a resource. For background activities, like live sync and reconciliation, this is not necessary at all. However, for listing accounts (Resources -> List resources -> Accounts) such a feature is really useful, if you have many accounts on your resource. In non-paged case, midPoint has to retrieve all accounts up to the currently displayed page (and it does not know how many accounts are there, so showing something like "Displaying 1 to 10 of 2 147 483 647 matching result."). When using paged search, it is sufficient to retrieve only those records, which are to be displayed. And midPoint knows how many objects are there.

How to enable paged search?

First of all, you have to use a connector that supports paged searches. For example, AD connector does this in its latest version of 1.4.1.20282, requiring latest ConnectorServer - 1.4.0.84.

Then, you have to configure the connector appropriately. This means:

1. Paged search is not compatible with the FilteredResultsHandler (a handler that provides additional filtering in the ConnId framework, in addition to filtering done in the connector). So, you have either to turn the filtered results handler off, or (if you are not sure whether this would not create any spurious results) enable it in validation mode. The configuration looks like this:

```
<icfc:resultsHandlerConfiguration>
    <icfc:enableFilteredResultsHandler>false</icfc:enableFilteredResultsHandler>
</icfc:resultsHandlerConfiguration>
```

or this:

```
<icfc:resultsHandlerConfiguration>
    <icfc:filteredResultsHandlerInValidationMode>true</icfc:filteredResultsHandlerInValidationMode>
    ...
</icfc:resultsHandlerConfiguration>
```

2. After that, you have to specify that paged searches are to be used. You can do this by creating **<pagedSearch>capability** - either at the level of resource or a resource object type. A minimal config looks like this:

At the level of resource:

```
<capabilities>
    <native> ... </native>
    <configured xmlns:cap="http://midpoint.evolveum.com/xml/ns/public/resource/capabilities-3">
        <cap:pagedSearch/>
    </configured>
</capabilities>
```

At the level of resource object type:

```
<schemaHandling>
    <objectType>
        <kind>account</kind>
        <displayName>Default Account</displayName>
        <default>true</default>
        ...
        <configuredCapabilities xmlns:cap="http://midpoint.evolveum.com/xml/ns/public/resource
/capabilities-3">
            <cap:pagedSearch/>
        </configuredCapabilities>
        ...
```

As paged searches require sorting of any kind (without sorting, the concept of a "page" has no meaning), you may define a default sorting - a field and an order. If you do not specify it, the connector may - or may not - supply its own default sorting. E.g. AD connector uses Common Name as a sorting criteria as the default. But you can use e.g. "sn", as indicated below:

```
<configuredCapabilities xmlns:cap="http://midpoint.evolveum.com/xml/ns/public/resource/capabilities-3">
    <cap:pagedSearch>
        <cap:defaultSortField xmlns:ri="http://midpoint.evolveum.com/xml/ns/public/resource/instance-3"
>ri:sn</cap:defaultSortField>
    </cap:pagedSearch>
 </configuredCapabilities>
```

midPoint (experimentally) supports paged searches since v3.0.1devel-576-g34f619c. **BEWARE: In v3.0.1devel-640-g6083c5f we have changed configuration from resource object type element <pagedSearches/> to the above capability-based config. Old style configuration is currently ignored. In future versions of midPoint (3.1 and above) any resource with <pagedSearches/> will be considered incorrect, i.e. <u>failing to be retrieved from the repository</u>! So if you use <pagedSearches> element, remove it as soon as possible.**

3. Finally, you have to tell midPoint to use paged search to estimate the number of objects. To do that, **<cap:countObjects><cap:simulate>pagedSearchEstimate</cap:simulate></cap:countObjects>** capability has to be used, e.g.

```
<configuredCapabilities xmlns:cap="http://midpoint.evolveum.com/xml/ns/public/resource/capabilities-3">
    <cap:pagedSearch/>
    <cap:countObjects>
        <cap:simulate>pagedSearchEstimate</cap:simulate>
    </cap:countObjects>
</configuredCapabilities>
```

Note that there is also a possibility to set simulate to "sequentialSearch". This method **does not** require paged searches. However, it is quite inefficient, because

a. it causes midPoint to fetch all resource objects in order to know their count;
b. when displaying objects in the middle of the list (e.g. those from 1001 to 1010), midPoint has to uselessly fetch first 1000 objects, just to get the required ones. Paged search is much more effective than this.

The number-of-objects estimation works since v3.0.1devel-765-gb7a0f58 (December 16, 2014).

Some AD-related notes

1. AD/Exchange connector implements paged search for objects that get retrieved from AD: users, orgs, groups. It does not support this kind of search for objects retrieved using PowerShell (all other Exchange-related object classes).
2. The sorting key must be indexed in AD. For example, icfs:name (a.k.a. __NAME__ a.k.a. distinguishedName) is not, neither is ad_container attribute. Among attributes that are indexed by default are cn and sn. The cn attribute is used by the connector if no sorting key is specified.
3. The implementation is based on VLV. So, in order to work, the VLV must not be disabled (see e.g. here).
4. If you still get the "**The server does not support the requested critical extension.**" error, consider
   a. applying kb977180-v2
   b. setting HKLM\System\CurrentControlSet\Services\NTDS\Parameters\DSA Heuristics to 000000000001 (string) - see this discussion thread - although beware if this fits into your environment!
5. If you encounter problems with queries with large result sets, you might want to tune your server's LDAP parameters.