Conflict resolution HOWTO

⚠ This feature is a bit experimental. Use with care.

If multiple operations are executed "at once" in parallel threads, some unexpected results might occur. See e.g.MID-4112: user has two assignments, both induce the same resource account. If these are deleted at the same time (each thread deleting one of them), the resource account is **not** deleted. The reason is that both operations see the object with one remaining assignment, and both think that the account must be kept.

A possible solution would be to enable only one operation at any given time. However, for deployments that have e.g. slow resources, this could lock out interactive operations for a considerable time: even for minutes. Therefore we have decided to implement "optimistic locking" (or "conflict detection" strategy). Any operation watches if there are any unexpected changes occurring to focus object it is operating on. And if it detects something, it tries to resolve the problem somehow.

How to configure

The following configuration snipped will enable basic conflict resolution for all kinds of focal objects (users, roles, orgs, services):

Configuration snippet to enable conflict resolution

```
<defaultObjectPolicyConfiguration>
   <conflictResolution>
      <action>recompute</action>
   </conflictResolution>
</defaultObjectPolicyConfiguration>
```

Basically it says: for all focal objects (note that "type" and "subtype" fields are not filled in) employ the recomputation-based conflict resolution. This means that if a conflict is detected, the object is recomputed after a small random delay.

Resolution actions

The following actions are available:

Action	Meaning
none	Nothing is to be done. This is the default action.
log	A warning is issued into the log file.
recompute	The focus object is recomputed.
fail	The whole operation fails. (This is to be used mainly for testing purposes, e.g. to check if conflict detection algorithm does not yield false positive results.)

Resolution attempts

When doing recomputation, it is possible (but not strictly necessary) to repeat the conflict resolution operation until no further conflicts are detected. In order to avoid repeated conflicts it is possible to configure two additional parameters:

Parameter	Meaning	Default value
maxAttempts	How many attempts to undertake at most.	1
delayUnit	What delay (in milliseconds) to introduce between attempts. Actual delay is taken as a random number between 0 and delayUnit x 2^{n-1}, where n is the number of conflict resolution attempt, starting at 1. (In other words, the potential delay is multiplied by two after each unsuccessful resolution attempt.)	5000

Currently we consider the default setting of maxAttempts = 1 to be adequate. Even if a conflict is detected during the recomputation, both threads should come to the same focal object state.

An example:

Fine-tuned conflict resolution

```
<defaultObjectPolicyConfiguration>
   <conflictResolution>
      <action>recompute</action>
      <maxAttempts>3</maxAttempts>
      <delayUnit>20000</delayUnit>
   </conflictResolution>
</defaultObjectPolicyConfiguration>
```