Grouper Subject API caching improvements in 2.4

Quick start

This is fixed in Grouper 2.4.0 patches api#15,16,17

You should do two things to get started:

1. configure a directory for the files that the cache stores to in subject.properties

```
########################################
## Subject caching
########################################

# if we are serializing to a file, enter a directory, else it is not used
# {valueType: "string"}
subject.cache.serializer.directory = /some/dir/for/cache
```

2. make sure all your subject identifiers are configured in your source. Check with this GSH command

```
groovy:000> SourceManager.getInstance().getSource("jdbc").getSubjectIdentifierAttributesAll();
===> [0:loginid]


If a subject identifier is not listed here (note, you dont need the subjectId listed), then add to the
subject.properties


# subject identifier to store in grouper's member table.  this is used to increase speed of loader and
perhaps for provisioning
# you can have up to max 1 subject identifier
subjectApi.source.jdbc.param.subjectIdentifierAttribute0.value = LOGINID


# subject identifier to store in grouper's member table.  this is used to increase speed of loader and
perhaps for provisioning
# you can have up to max 1 subject identifier
subjectApi.source.jdbc.param.subjectIdentifierAttribute1.value = eppn
```

Design

Grouper will keep a list of subjects in memory for certain sources (e.g. not groups since there is security involved and it is more dynamic). The list size will be configurable. This will cache lookups by id or identifer, not freeform searches.

Each object in memory will have a structure:

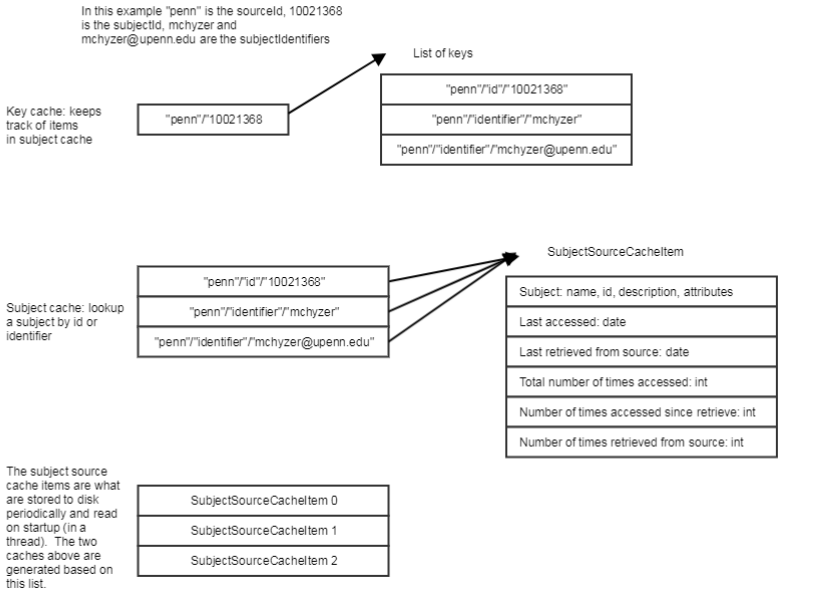
1. The subject
2. ID
3. Identifiers
4. Last retrieved
5. Last accessed
6. Number of times accessed since last retrieved

Periodically this set will be serialized and stored to disk (e.g. every 30 minutes). Grouper would need this configured with a place to store. Multiple grouper JVMs could share the same folder if the subject sources are configured identically, though you probably dont want to do this. Each read and write files with unique names, and each type of grouper JVM (UI/WS/Daemon), will have different subjects in their cache... On startup it will read this file. Note, it will store another file before deleting the current file in case the JVM abruptly stops during processing.

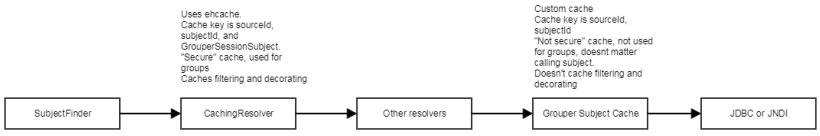
A background thread will refresh old often-used subjects in the cache that are too old (configurable).

There should be a documented way to bypass the cache (e.g. when the cache is looking up subjects).

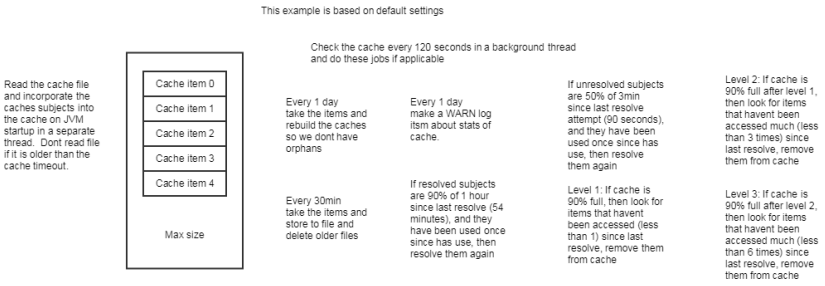
This will not cache when a subject is not found.



This is how the cache fits in with existing cache



This is how resolving and deleting of the cache works



Configuration

in subject.properties

```
########################################
## Subject caching
```

```
########################################

# There are various caches for subjects.
# This cache will alleviate calls to databases and ldaps.
# {valueType: "string"}
subject.cache.enable = true

# implementation of a serializer for the cache
# {valueType: "class", mustExtendClass: "edu.internet2.middleware.grouper.subj.cache.SubjectSourceSerializer"}
subject.cache.serializer = edu.internet2.middleware.grouper.subj.cache.SubjectSourceSerializerFile

# if we are serializing to a file, enter a directory, else it is not used
# {valueType: "string"}
subject.cache.serializer.directory =

# max subjects in cache
# {valueType: "integer"}
subject.cache.maxElementsInMemory = 200000

# if the cache should auto refresh in background
# {valueType: "boolean"}
subject.cache.autoRefresh = true

# Sets the time to live cycle for an element before it expires or gets refreshed.  Default 1 hour
# {valueType: "integer", required: true}
subject.cache.timeToLiveSeconds = 3600

# Resolve subjects again if necessary, after this percent of time to live
# {valueType: "integer", required: true}
subject.cache.timeToLivePercentageToResolveSubjectsIfNecessary = 90

# Sets the time to live cycle for an element which is not found before it expires or gets refreshed.  Default 3
minute
# {valueType: "integer", required: true}
subject.cache.timeToLiveNotFoundSeconds = 180

# Resolve subjects again if necessary, after this percent of time to live for not found subjects.
# note if the time to live for not found subjects is low, this has to be low too so there is time to resolve...
# {valueType: "integer", required: true}
subject.cache.timeToLiveNotFoundPercentageToResolveSubjectsIfNecessary = 50

# if the subject has been used at least this many times in the last cycle then auto refresh the subject
# or else just remove the subject.  in other words if the subject is used once (default) in the time to live
(default is 1 hour),
# then dont remove from cache.  This will make room for new subjects in cache
# set to 0 to refresh all
# {valueType: "integer", required: true}
subject.cache.minUseInCycleToAutoRefresh = 1

# if the subject has been used at least this many times in the last cycle then dont delete unless needed.
# If the cache is 90% full, and all subjects that havent been used once have been deleted, then in the second
# level delete all subjects that were used less than 3 times, to make room for new subjects in cache
# {valueType: "integer", required: true}
subject.cache.minUseLevel2ToNotDelete = 3

# if the subject has been used at least this many times in the last cycle then dont delete unless needed.
# if the cache is 90% full, and the level 1 pass and level 2 pass have been completed, and the cache is still
# 90% full, then delete subjects that were used less than 6 times, to make room for new subjects.
# {valueType: "integer", required: true}
subject.cache.minUseLevel3ToNotDelete = 6

# will log stats of the cache to WARN from edu.internet2.middleware.grouper.subj.cache.SubjectSourceCache
# -1 to not log.  Might want to write daily: 86400
# {valueType: "integer"}
subject.cache.logStatsSeconds = 86400

# exclude these sourceIds from cache, comma separated
# {valueType: "string", multiple: true}
subject.cache.excludeSourceIds = $$subjectApi.source.g_gsa.id$$

# this helps configure the subject source correctly.  if a subject is found by identifier, and that attribute is
```

```
# not mapped in the subject source, log at warn level so the admin can change the subject source so grouper
knows the attribute is an identifier.
# {valueType: "boolean"}
subject.cache.logWarnIfIdentiferNotConfigured = true

# sleep this many seconds in between looking for items to delete (if cache is full).  wont run more than once
per minute
# {valueType: "integer", required: true}
subject.cache.waitSecondsBetweenSweepforDeletes = 60

# dont sweep cache unless the cache is this percent full.  note this operation is quick but dont overdo it
# sweeping will delete items in cache which are expired and not used often, or if that doesnt help, then if
they are expired
# {valueType: "integer"}
subject.cache.dontSweepCacheForDeletesUnlessCacheIsThisPercentFull = 90

# store a copy to external storage after this many seconds
# {valueType: "integer"}
subject.cache.storeToStorageAfterThisManySeconds = 1800

# rebuild cache to make consistent after this many seconds.  this does not resolve all subjects, just internal
# {valueType: "integer"}
subject.cache.rebuildCacheAfterThisManySeconds = 86400

# clone subjects out of cache in case they are edited
# {valueType: "boolean"}
subject.cache.cloneSubjectsOnReturn = true

# sleep this many seconds in between thread runs (which checks for delete, resolved, file store, etc)
# {valueType: "integer"}
subject.cache.sleepSecondsInBetweenThreadRuns = 120
```

Logging

To see stats, make sure grouper logs to WARN, and by default you will get daily stats for each JVM

```
2019-01-14 22:55:57,157: [main] WARN  SubjectSourceCache.logStats(627) -  - date: 2019_01_14, subjectsInCache:
1, cacheKeysIdsIdentifiers: 1, cacheHitsIndividual: 0, microsPerCacheHit: 0, cacheMissesIndividual: 1,
microsPerCacheMissIndividual: 1601, cacheMissesForList: 0, microsPerCacheMissForList: 0, cacheMissesResolved:
0, cacheMissesUnresolved: 1, cacheRemoves: 0, cacheHitsSinceLastRetrieve: 0, cacheHitsTotalOfItemsInCache: 1,
cacheSizeSubjectsResolved: 0, cacheSizeSubjectsUnresolved: 1, numberOfSubjectsRefreshedInBackground: 0,
refreshQueriesInBackground: 0, microsPerBackgroundRefreshQuery: 0, cacheRefreshesTotalOfItemsInCache: 1,
itemsWithNoAccessSinceLastRefresh: 1, itemsWith1AccessSinceLastRefresh: 0, itemsWith3AccessSinceLastRefresh: 0,
itemsWith10AccessSinceLastRefresh: 0, itemsWith20AccessSinceLastRefresh: 0, itemsWith1Refresh: 1,
itemsWith3Refresh: 0, itemsWith10Refresh: 0, itemsWith20Refresh: 0, itemsWith50Refresh: 0,
itemsWith1AccessTotal: 1, itemsWith3AccessTotal: 0, itemsWith10AccessTotal: 0, itemsWith20AccessTotal: 0,
itemsWith50AccessTotal: 0
```

Set this in log4j.properties to see detailed logs

```
log4j.logger.edu.internet2.middleware.grouper.subj.cache = DEBUG
```

Background

For a long time Grouper has been distributed with some default ehcache settings which have appeared to be ok. Some have had to run Grouper on a laptop to demonstrate its capabilities with the dockerized version of Grouper. Using an OracleDB as the subject source and everything else running local it was found to be slow and after a bit of poking around it would appear changes to the subject caching can have a significant impact on the overall grouper performance - especially where integrations (changelog consumers) are involved. It may well be changes to the caching for the grouper DB might have equally dramatic impact, but for now, some capabilities for better managing the subject data would be appreciated.

Finding all the objects for ehcache to better manage subject caching is not something the average grouper admin can determine. Either grouper clearly defines which objects are subject source related (doing so is still probably a useful effort independent of solution) or some generalized config for subject caching should be provided. It has also be discussed about the need for some disk caching of subject data to survive server (or even GSH) restarts. If disk caching is employed, then some mechanism to "clear the cache" would also be needed for troubleshooting issues. As part of being to calculate in-memory caching needs, the maximum size of a subject to cache would need to be exposed. This value could be presented at server (or GSH) startup.

Given the above discussion the following items need exploration:

1. defining which objects are subject source related
    a. co-locating these objects in the cache config properties file
    b. note: queries to the grouper_members table too
    c. do we only want to cache certain subject source (i.e. maybe not cache the groups subject source as strongly?)
        i. Chris - take a look at the config below. Does any of that impact groups subject source? If so, then it may come into scope for this effort.
2. provide max size of a subject entry to cache to enable better estimates of memory requirements.
3. provide options for disk caching of objects to survive restarts.
    a. ability to enable/disable disk caching on a running system
    b. ability to clear the disk cache of a running system
    c. we do have overflow to disk, and that defaults to the tmp dir, and after a restart it will make new files. maybe we should rethink that approach or add a param that says "we have a unique directory for this jvm, so dont create new cache files on restart"
        i. we might want to discuss further with Hubing and Caskey from the Container perspective... should these cache files be inside or outside the container? On restart, should cache files of a certain age be used or discarded?
4. the default cache config for grouper subjects should be for 200K subjects and 1 hour caching
    a. disk caching - if determined to be desirable - should default to on.
5. should there be a way to download the entire subject source periodically? (e.g. for SQL sources)
6. should there be a way to trigger changes to the cache with a real time notification like the real time loader?
7. can we determine the avg size in memory of one subject? and the size of the entire cache and number of items in the cache?

As part of the experiment to improve subject caching, the following are the entries used for the grouper.cache.properties file. Note, you shouldnt change these settings, the above work takes the place of these...

grouper.cache.properties

```
cache.name.RegistrySubject.timeToIdleSeconds = 2700
cache.name.RegistrySubject.timeToLiveSeconds = 2700
cache.name.RegistrySubjectAttribute.timeToIdleSeconds = 2700
cache.name.RegistrySubjectAttribute.timeToLiveSeconds = 2700
cache.name.attr_finder_AttributeDefNameFinder_findByNameCache.timeToIdleSeconds = 2700
cache.name.attr_finder_AttributeDefNameFinder_findByNameCache.timeToLiveSeconds = 2700
cache.name.beans_HooksContext_subjectInGroupCache.timeToIdleSeconds = 2700
cache.name.beans_HooksContext_subjectInGroupCache.timeToLiveSeconds = 2700
cache.name.entity_EntitySubject_EntityAttributeIdCache.timeToIdleSeconds = 2700
cache.name.entity_EntitySubject_EntityAttributeIdCache.timeToLiveSeconds = 2700
cache.name.externalSubjects_ExternalSubject.timeToIdleSeconds = 2700
cache.name.externalSubjects_ExternalSubject.timeToLiveSeconds = 2700
cache.name.externalSubjects_ExternalSubjectAttribute.timeToIdleSeconds = 2700
cache.name.externalSubjects_ExternalSubjectAttribute.timeToLiveSeconds = 2700
cache.name.externalSubjects_ExternalSubjectConfig_autoaddConfigCache.timeToIdleSeconds = 2700
cache.name.externalSubjects_ExternalSubjectConfig_autoaddConfigCache.timeToLiveSeconds = 2700
cache.name.externalSubjects_ExternalSubjectConfig_configCache.timeToIdleSeconds = 2700
cache.name.externalSubjects_ExternalSubjectConfig_configCache.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByAttributeDefId.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByAttributeDefId.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindById.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindById.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindByUuidOrKey.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindByUuidOrKey.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignValueDAO_FindByAttributeAssignId.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignValueDAO_FindByAttributeAssignId.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByAttributeDefNameIdSecure.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByAttributeDefNameIdSecure.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindById.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindById.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByUuidsSecure.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByUuidsSecure.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByNameCache.timeToIdleSeconds = 2700
```

```
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByNameCache.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidsSecure.timeToIdleSeconds = 2700
cache.name.externalSubjects_ExternalSubjectConfig_configCache.timeToIdleSeconds = 2700
cache.name.externalSubjects_ExternalSubjectConfig_configCache.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByAttributeDefId.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByAttributeDefId.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignActionDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindById.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindById.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindByUuidOrKey.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignDAO_FindByUuidOrKey.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignValueDAO_FindByAttributeAssignId.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeAssignValueDAO_FindByAttributeAssignId.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByAttributeDefNameIdSecure.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByAttributeDefNameIdSecure.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindById.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindById.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByUuidsSecure.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefDAO_FindByUuidsSecure.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByNameCache.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByNameCache.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidsSecure.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefNameDAO_FindByUuidsSecure.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefScopeDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AttributeDefScopeDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AuditEntryDAO_FindByActingUser.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AuditEntryDAO_FindByActingUser.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AuditTypeDAO_FindByCategory.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AuditTypeDAO_FindByCategory.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3AuditTypeDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3AuditTypeDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3ChangeLogEntryDAO_FindBySequenceNumber.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3ChangeLogEntryDAO_FindBySequenceNumber.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3CompositeDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3CompositeDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3FieldDAO_FindByUuidOrName.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3FieldDAO_FindByUuidOrName.timeToLiveSeconds = 2700
cache.name.internal_dao_hib3_Hib3MemberDAO_FindBySubject.timeToIdleSeconds = 2700
cache.name.internal_dao_hib3_Hib3MemberDAO_FindBySubject.timeToLiveSeconds = 2700
cache.name.subj_CachingResolver_Find.timeToIdleSeconds = 2700
cache.name.subj_CachingResolver_Find.timeToLiveSeconds = 2700
cache.name.subj_CachingResolver_FindAll.timeToIdleSeconds = 2700
cache.name.subj_CachingResolver_FindAll.timeToLiveSeconds = 2700
cache.name.subj_CachingResolver_FindByIdOrIdentifier.timeToIdleSeconds = 2700
cache.name.subj_CachingResolver_FindByIdOrIdentifier.timeToLiveSeconds = 2700
cache.name.subj_CachingResolver_FindByIdentifier.timeToIdleSeconds = 2700
cache.name.subj_CachingResolver_FindByIdentifier.timeToLiveSeconds = 2700
cache.name.subj_CachingResolver_FindPage.timeToIdleSeconds = 2700
cache.name.subj_CachingResolver_FindByIdentifier.timeToLiveSeconds = 2700
cache.name.subj_CachingResolver_FindPage.timeToIdleSeconds = 2700
cache.name.subj_CachingResolver_FindPage.timeToLiveSeconds = 2700
```