Syncing groups on demo server

Synching Groups on the Demo Server

Basically there are two 2.0 Groupers on the demo server. Note that only external subjects are synced (not group members, or GrouperSystem, etc). The source groups are world readable/writable so everyone can experiment. The destination groups are world readable (should be readonly). There are three groups synced across, in the three different sync strategies (note that in real life, the diffs are still every minute, but the crons for full refreshes will probably be less frequent (daily?) though are configurable):

1. push (cron'ed for every 5 minutes to do a full sync). Source group. Destination group
2. incremental_push (will do diffs every minute, and cron'ed for every 5 minutes to do a full sync). Source group. Destination group
3. pull (cron'ed for every 5 minutes to do a full sync). Destination group. Source group

Groups on grouperdemo v2_0_0:

```
gsh 0% grouperSession = GrouperSession.startRootSession();
gsh 1% new GroupSave(grouperSession).assignName("groupSync_v2_0_0:sourcePushGroup").
assignCreateParentStemsIfNotExist(true).save();
gsh 2% new GroupSave(grouperSession).assignName("groupSync_v2_0_0:sourcePushIncrementalGroup").
assignCreateParentStemsIfNotExist(true).save();
gsh 3% new GroupSave(grouperSession).assignName("groupSync_v2_0_0:destinationPullGroup").
assignCreateParentStemsIfNotExist(true).save();
gsh 4% addSubject("remoteUser", "application", "remoteUser");
gsh 5% grantPriv("groupSync_v2_0_0:sourcePushGroup", "remoteUser", AccessPrivilege.READ);
gsh 6% grantPriv("groupSync_v2_0_0:sourcePushIncrementalGroup", "remoteUser", AccessPrivilege.READ);
gsh 7% grantPriv("groupSync_v2_0_0:destinationPullGroup", "remoteUser", AccessPrivilege.READ);
gsh 8% grantPriv("groupSync_v2_0_0:destinationPullGroup", "remoteUser", AccessPrivilege.UPDATE);

[mchyzer@i2midev1 ~]$ sudo htpasswd /etc/httpd/conf.d/users.pass remoteUser
```

Groups on grouperdemo v2_0_0a:

```
gsh 0% grouperSession = GrouperSession.startRootSession();
gsh 1% new GroupSave(grouperSession).assignName("groupSync_v2_0_0a:destinationPushGroup").
assignCreateParentStemsIfNotExist(true).save();
gsh 2% new GroupSave(grouperSession).assignName("groupSync_v2_0_0a:destinationPushIncrementalGroup").
assignCreateParentStemsIfNotExist(true).save();
gsh 3% new GroupSave(grouperSession).assignName("groupSync_v2_0_0a:sourcePullGroup").
assignCreateParentStemsIfNotExist(true).save();
gsh 4% addSubject("remoteUser", "application", "remoteUser");gsh 5% grantPriv("groupSync_v2_0_0a:
destinationPushGroup", "remoteUser", AccessPrivilege.READ);
gsh 6% grantPriv("groupSync_v2_0_0a:destinationPushGroup", "remoteUser", AccessPrivilege.UPDATE);
gsh 7% grantPriv("groupSync_v2_0_0a:destinationPushIncrementalGroup", "remoteUser", AccessPrivilege.READ);
gsh 8% grantPriv("groupSync_v2_0_0a:destinationPushIncrementalGroup", "remoteUser", AccessPrivilege.UPDATE);
gsh 6% grantPriv("groupSync_v2_0_0a:sourcePullGroup", "remoteUser", AccessPrivilege.READ);
```

Test a client:

```
grouper.client.properties:
grouperClient.webService.url = https://grouperdemo.internet2.edu/grouper-ws_v2_0_0/servicesRest
grouperClient.webService.login = remoteUser
grouperClient.webService.password = **********

C:\temp\grouperclient>java -jar grouperClient.jar --operation=addMemberWs --groupName=groupSync_v2_0_0:
sourcePushGroup --subjectIds=remoteUser
Index 0: success: T: code: SUCCESS: remoteUser

C:\temp\grouperclient>java -jar grouperClient.jar --operation=getMembersWs --groupNames=groupSync_v2_0_0:
sourcePushGroup
GroupIndex 0: success: T: code: SUCCESS: group: groupSync_v2_0_0:sourcePushGroup: subjectIndex: 0: remoteUser
```

Setup a sync:

grouper.properties:

```
#####################################
## Grouper client connections
## if this grouper needs to talk to another grouper, this is the client connection information
#####################################


# id of the source, should match the part in the property name
grouperClient.grouperDemo_v2_0_0a.id = grouperDemo_v2_0_0a

# url of web service, should include everything up to the first resource to access
# e.g. https://groups.school.edu/grouperWs/servicesRest
grouperClient.grouperDemo_v2_0_0a.properties.grouperClient.webService.url = https://grouperdemo.internet2.edu
/grouper-ws_v2_0_0a/servicesRest

# login ID
grouperClient.grouperDemo_v2_0_0a.properties.grouperClient.webService.login = remoteUser

# password for shared secret authentication to web service
# or you can put a filename with an encrypted password
grouperClient.grouperDemo_v2_0_0a.properties.grouperClient.webService.password = /opt/grouper/2.0.0/pass
/remoteUser.pass

# client version should match or be related to the server on the other end...
grouperClient.grouperDemo_v2_0_0a.properties.grouperClient.webService.client.version = v2_0_000

# this is the subject to act as local, if blank, act as GrouperSystem, specify with SubjectFinder packed
string, e.g.
# subjectIdOrIdentifier  or  sourceId:::subjectId  or  ::::subjectId  or  sourceId::::::
subjectIdentifier  or  ::::::subjectIdentifier
# sourceId::::::::subjectIdOrIdentifier  or  ::::::::subjectIdOrIdentifier
grouperClient.grouperDemo_v2_0_0a.localActAsSubject = remoteUser


# the id of this source, generally the same as the name in the property name.  This is mandatory
grouperClient.grouperDemo_v2_0_0a.source.externalUser.id = grouperExternal

# the part between "grouperClient.someOtherSchool.source." and ".id" links up the configs,
# in this case, "jdbc", make sure it has no special chars.  sourceId can be blank if you dont want to specify
grouperClient.grouperDemo_v2_0_0a.source.externalUser.local.sourceId = grouperExternal

# this is the identifier that goes between them, it is "id" or an attribute name.  subjects without this
attribute will not be processed
grouperClient.grouperDemo_v2_0_0a.source.externalUser.local.read.subjectId = identifier

# this is the identifier to lookup to add a subject, should be "id" or "identifier" or "idOrIdentifier"
grouperClient.grouperDemo_v2_0_0a.source.externalUser.local.write.subjectId = identifier

# sourceId of the remote system, can be blank
```

```
grouperClient.grouperDemo_v2_0_0a.source.externalUser.remote.sourceId = grouperExternal

# this is the identifier that goes between them, it is "id" or an attribute name.  subjects without this
attribute will not be processed
grouperClient.grouperDemo_v2_0_0a.source.externalUser.remote.read.subjectId = identifier

# this is the identifier to lookup to add a subject, should be "id" or "identifier" or "idOrIdentifier"
grouperClient.grouperDemo_v2_0_0a.source.externalUser.remote.write.subjectId = identifier



#####################################
## Sync to/from another grouper
## Only sync one group to one other group, do not sync one group to
## two report groupers.  If you need to do this, add the group to another group
#####################################

# we need to know where our
# connection name in grouper client connections above
syncAnotherGrouper.sourcePushGroup.connectionName = grouperDemo_v2_0_0a

# incremental  or  push  or   pull  or  incremental_push.  Note, incremental push is cron'ed and incremental
(to make sure no discrepancies arise)
syncAnotherGrouper.sourcePushGroup.syncType = push

# quartz cron  to schedule the pull or push (incremental is automatic as events happen) (e.g. 5am daily)
syncAnotherGrouper.sourcePushGroup.cron =  0 0/5 * * * ?


# local group which is being synced
syncAnotherGrouper.sourcePushGroup.local.groupName = groupSync_v2_0_0:sourcePushGroup

# remote group at another grouper which is being synced
syncAnotherGrouper.sourcePushGroup.remote.groupName = groupSync_v2_0_0a:destinationPushGroup

# if subjects are external and should be created if not exist
syncAnotherGrouper.sourcePushGroup.addExternalSubjectIfNotFound = true

#############

# we need to know where our
# connection name in grouper client connections above
syncAnotherGrouper.sourcePushIncrementalGroup.connectionName = grouperDemo_v2_0_0a

# incremental  or  push  or   pull  or  incremental_push.  Note, incremental push is cron'ed and incremental
(to make sure no discrepancies arise)
syncAnotherGrouper.sourcePushIncrementalGroup.syncType = incremental_push

# quartz cron  to schedule the pull or push (incremental is automatic as events happen) (e.g. 5am daily)
syncAnotherGrouper.sourcePushIncrementalGroup.cron =  0 0/5 * * * ?

# local group which is being synced
syncAnotherGrouper.sourcePushIncrementalGroup.local.groupName = groupSync_v2_0_0:sourcePushIncrementalGroup

# remote group at another grouper which is being synced
syncAnotherGrouper.sourcePushIncrementalGroup.remote.groupName = groupSync_v2_0_0a:
destinationPushIncrementalGroup

# if subjects are external and should be created if not exist
syncAnotherGrouper.sourcePushIncrementalGroup.addExternalSubjectIfNotFound = true

#############

# we need to know where our
# connection name in grouper client connections above
syncAnotherGrouper.destinationPullGroup.connectionName = grouperDemo_v2_0_0a

# incremental  or  push  or   pull  or  incremental_push.  Note, incremental push is cron'ed and incremental
(to make sure no discrepancies arise)
syncAnotherGrouper.destinationPullGroup.syncType = pull
```

```
# quartz cron  to schedule the pull or push (incremental is automatic as events happen) (e.g. 5am daily)
syncAnotherGrouper.destinationPullGroup.cron =  0 0/5 * * * ?

# local group which is being synced
syncAnotherGrouper.destinationPullGroup.local.groupName = groupSync_v2_0_0:destinationPullGroup

# remote group at another grouper which is being synced
syncAnotherGrouper.destinationPullGroup.remote.groupName = groupSync_v2_0_0a:sourcePullGroup

# if subjects are external and should be created if not exist
syncAnotherGrouper.destinationPullGroup.addExternalSubjectIfNotFound = true
```

sadf