v2.2 Upgrade Instructions from v2.1

- Grouper v2.2.1 upgrade to v2.2.2 example
- Grouper v2.2.2 upgrade from v2.2.1 on the demo server

Upgrading to Grouper 2.2 from Grouper 2.1

Using the Grouper Upgrader can simplify your upgrade process. Here is a movie demonstrating the Grouper upgrader. The upgrader can upgrade an installed env of the API, UI, WS, client, PSP, etc. If you dont have a build script to manage multiple envs, you might want to use the upgrader.

These instructions describe how you can upgrade to Grouper 2.2 from 2.1. Note that if you are upgrading from 1.6 or 2.0, there are special notes below on what to do differently.

2014/09/01: for 2.2.1+ For the UI, edit the tomcat server.xml <Connectors to have uri encoding of utf8: <Connector URIEncoding="UTF-8"

2015/01/18: If you have installed UI patches before this date, you should edit the grouperPatchStatus.properties file (in WEB-INF). Change two instances of grouper_v2_2_1_ui_patch_3 to grouper_v2_2_1_api_patch_2. These are the same patch and it belongs in the API not UI.

2015/09/30: Make sure to run the grouperInstaller of the version of grouper you are upgrading from to revert all patches before upgrading. Note, If reverting patches doesn't work, and you skipped that part in an upgrade, then before new patches are applied, look through the WEB-INF/classes dir for classfiles, and if you have never overridden or coded your own, then delete all classfiles in the subdirs. If you have, then keep yours, and delete the others.

Important Changes in Grouper 2.2 that impact the upgrade

Configuration Files: In order to make Grouper more easily deployable across environments, and more easily upgradable, Grouper now has configuration overlay files, and the ability to use expression language in config file entries. **It is highly recommended that as you upgrade to Grouper 2.2, you restructure your configuration files to take advantage of the configuration overlay files.**

Here is how it works:

- There is default configuration file (called the base file), and there is an override (overlay) file that includes only the changes from the default. Several, but not all, of Grouper's configuration files now use this structure.
- In the past, the grouper.properties file would contain all the default properties along with your changes.
- Starting with Grouper 2.2, the grouper.properties file should only contain your changes from the base file. Note: "grouper.properties" is used as an example file name here, there are several other files such as grouper-ws.properties, etc.)
- If you don't know what your changes were from the base configuration, you should diff the file with the example file from that release (i.e. diff grouper.properties grouper.example.properties).
- The results of the diff should go into the override file.
- The base file should be used as provided.
- See Grouper configuration overlay for more details and which configuration files use this new method.
- The sources.xml file does not use the overlay method. Also, the sources.xml file from 2.1 is compatible with 2.2.

Legacy Style Attributes: The legacy style attributes and group types are no longer part of Grouper. Those legacy APIs still work though; they just use the newer attribute framework. This upgrade process includes migrating those legacy attributes and group types to the newer attribute framework. Read more about the legacy attribute and group type migration.

- If you are unsure about the legacy group types that you have, you can look in the grouper_types table. The group types "base", "naming", and "attributeDef" were all internal and no longer apply. If you have custom code that tries to find those group types now, you will receive an exception. The rest will be migrated including any built-in group types that you may be using (e.g. addIncludeExclude, requireInGroups, grouperLoader). Also note that the legacy group types are visible in the admin UI when viewing/updating a group.

- If you are unsure about the legacy attributes that you have, you can look in the grouper_fields table for rows where the type column is set to "attribute". All of these attributes will be migrated to the newer attribute framework.

New Privileges: New privileges have been added to determine who has read and update access to attributes based on what the attribute is assigned to. The new privileges are: groupAttrRead, groupAttrUpdate, stemAttrRead, stemAttrUpdate, attrDefAttrRead, and attrDefAttrUpdate. These privileges are not set during the upgrade, so if your users need read or update access to attributes and they are not admins of those objects, then the appropriate privilege would need to be assigned. This applies to attributes originally created using the newer attribute framework as well as attributes using the old attribute framework that would be migrated during this upgrade. See the section of the Attribute Framework page titled New Privileges for Attributes in Grouper 2.2+

GrouperAll (aka EveryEntity): Starting from 2.2.1, GrouperAll isn't allowed to be granted ADMIN, UPDATE, or GROUP_ATTR_UPDATE privileges on groups. It is also not allowed to become a member of a group. If you are upgrading from 2.1 (or before) directly to 2.2.1+ (thus skipping 2.2.0), then the upgrade steps below will remove these privileges and memberships. If you are upgrading from 2.2.0 to 2.2.1+, then run "gsh misc /postGrouper2_2_1Upgrade.gsh". The upgrader also takes care of this.

Other items before upgrading

1. You may want to have your DBAs make sure you are not close to running out of tablespace. In general, it may be useful to have your DBAs available when you upgrade.
2. If you have views that other systems use, you could replace them as tables before beginning.

3. If you have other systems using Grouper, you could temporarily disable them.
4. If you have a large number of groups/folders, it would probably be a good idea to add temporary indexes on grouper_change_log_entry_temp.string01 and grouper_change_log_entry_temp.string02 to help speed up Step 9 below. You can drop the indexes after the upgrade. (The index creation will probably later be added automatically during the upgrade but it does not do that right now.)
5. The upgrade adds new unique indexes to the point in time audit tables. The index creations may fail if you have duplicate rows, which could have happened if you ever had multiple instances of the Grouper daemon running at the same time. Check to make sure that the following queries don't return anything before starting the upgrade. If they do, then the duplicate rows would need to be cleaned up first.

```
select source_id, start_time from grouper_pit_attr_assn_actn group by source_id, start_time having count
(*) > 1;
select source_id, start_time from grouper_pit_attr_assn_actn_set group by source_id, start_time having
count(*) > 1;
select source_id, start_time from grouper_pit_attr_assn_value group by source_id, start_time having count
(*) > 1;
select source_id, start_time from grouper_pit_attr_def_name group by source_id, start_time having count
(*) > 1;
select source_id, start_time from grouper_pit_attr_def_name_set group by source_id, start_time having
count(*) > 1;
select source_id, start_time from grouper_pit_attribute_assign group by source_id, start_time having
count(*) > 1;
select source_id, start_time from grouper_pit_attribute_def group by source_id, start_time having count
(*) > 1;
select source_id, start_time from grouper_pit_fields group by source_id, start_time having count(*) > 1;
select source_id, start_time from grouper_pit_group_set group by source_id, start_time having count(*) >
1;
select source_id, start_time from grouper_pit_groups group by source_id, start_time having count(*) > 1;
select source_id, start_time from grouper_pit_members group by source_id, start_time having count(*) > 1;
select source_id, start_time from grouper_pit_memberships group by source_id, start_time having count(*)
> 1;
select source_id, start_time from grouper_pit_role_set group by source_id, start_time having count(*) >
1;
select source_id, start_time from grouper_pit_stems group by source_id, start_time having count(*) > 1;
```

Upgrade Steps

1. If you are upgrading from 1.6 or 2.0, take a look at item #1 in the 2.1 upgrade instructions.
2. You should get v2.2 versions of the Grouper API, Grouper UI, Grouper WS, Grouper Daemon, etc. from the Grouper Downloads page. You will need to merge configuration files and JARs. See the v2.2 change log for more information. Also keep in mind that some of the configuration files are now handled differently via configuration overlay. The rest of this document focuses on upgrading the database.
3. First you may want to analyze your tables to help speed up the upgrade. Analyze your tables.
4. Stop the Grouper Daemon. Once you prevent users from making updates to your Grouper instance, run the changeLogTempToChangeLog daemon to clear out the temp changelog using your existing v2.1 API. Here's an example using GSH.

```
gsh 0% loaderRunOneJob("CHANGE_LOG_changeLogTempToChangeLog")
```

5. Before performing any upgrade steps, export your Grouper registry. Options include performing a database backup (recommended) or using the XML Export utility in Grouper (not recommended since certain features may not get exported).
6. Using the 2.2 API, perform a registry check using GSH to create an SQL file that will contain the DDL to update your database. To do this, run: gsh -registry -check Note you may need to increase memory. For instance..

```
$ export MEM_MAX=2000m
$ ./bin/gsh.sh -registry -check
Using GROUPER_HOME: /opt/grouper
Using GROUPER_CONF: /opt/grouper/conf
Using JAVA: java
using MEMORY: 64m-2000m
Grouper starting up: version: 2.2.0, build date: 2014/05/23 13:16:59, env: <no label configured>
grouper.properties read from: /opt/grouper/conf/grouper.properties
Grouper current directory is: /opt/grouper
log4j.properties read from:   /opt/grouper/conf/log4j.properties
Grouper is logging to file:   /opt/grouper/logs/grouper_error.log, at min level WARN for package: edu.
internet2.middleware.grouper, based on log4j.properties
grouper.hibernate.properties: /opt/grouper/conf/grouper.hibernate.properties
grouper.hibernate.properties: sa@jdbc:hsqldb:hsql://localhost:9001/grouper
sources.xml read from:        /opt/grouper/conf/sources.xml
sources.xml groupersource id: g:gsa
sources.xml groupersource id: grouperEntities
sources.xml jdbc source id:   jdbc: GrouperJdbcConnectionProvider
This db user 'sa' and url 'jdbc:hsqldb:hsql://localhost:9001/grouper' are allowed to be changed in the
grouper.properties
Continuing...
Grouper ddl object type 'Grouper' has dbVersion: 26 and java version: 28
Grouper database schema DDL requires updates
(should run script manually and carefully, in sections, verify data before drop statements, backup
/export important data before starting, follow change log on confluence, dont run exact same script in
multiple envs - generate a new one for each env),
script file is:
/opt/grouper/ddlScripts/grouperDdl_20140523_13_19_30_162.sql
Note: this script was not executed due to option passed in
To run script via gsh, carefully review it, then run this:
gsh -registry -runsqlfile /opt/grouper/ddlScripts/grouperDdl_20140523_13_19_30_162.sql
```

   a. In this example above, an SQL script called /opt/grouper/ddlScripts/grouperDdl_20140523_13_19_30_162.sql was created.
   b. Postgres only - If using postgres, you should see foreign keys being dropped at the top of the script. If not, try setting the ddlutils.schema grouper.properties setting and run again. If you still don't see foreign keys being dropped at the top of the script, manually drop all foreign keys before running the script.
   c. Postgres and hsql only - You should backup any non grouper views that depend on Grouper views, run the grouper script (which deletes those views due to drop view cascade), and then you should recreate those non grouper views.

7. Run the SQL script. To do this, run: gsh -registry -runsqlfile /path/to/sql/file.sql For instance..

```
$ ./bin/gsh.sh -registry -runsqlfile /opt/grouper/ddlScripts/grouperDdl_20140523_13_19_30_162.sql
Using GROUPER_HOME: /opt/grouper
Using GROUPER_CONF: /opt/grouper/conf
Using JAVA: java
using MEMORY: 64m-2000m
This db user 'sa' and url 'jdbc:hsqldb:hsql://localhost:9001/grouper' are allowed to be changed in the
grouper.properties
Continuing...
Script was executed successfully

Grouper starting up: version: 2.2.0, build date: 2014/05/23 13:16:59, env: <no label configured>
grouper.properties read from: /opt/grouper/conf/grouper.properties
Grouper current directory is: /opt/grouper
log4j.properties read from:   /opt/grouper/conf/log4j.properties
Grouper is logging to file:   /opt/grouper/logs/grouper_error.log, at min level WARN for package: edu.
internet2.middleware.grouper, based on log4j.properties
grouper.hibernate.properties: /opt/grouper/conf/grouper.hibernate.properties
grouper.hibernate.properties: sa@jdbc:hsqldb:hsql://localhost:9001/grouper
sources.xml read from:        /opt/grouper/conf/sources.xml
sources.xml groupersource id: g:gsa
sources.xml groupersource id: grouperEntities
sources.xml jdbc source id:   jdbc: GrouperJdbcConnectionProvider
```

Note that if one of the SQL statements in the script fails, the process will abort leaving the rest of the SQL statements from executing. If this happens, in most cases, you can't just re-run the full script since re-executing some of the DDL changes that previously succeeded would fail now (e.g. dropping a view or constraint that was previously dropped successfully.) You could edit the script to remove the statements that previously succeeded in order to re-execute the statement that failed and the ones after it. Or you can run the previous step again to generate a new SQL script.

8. If you are upgrading from 1.6, take a look at item #14 in the 2.1 upgrade instructions.
9. Now that the DDL updates have been made, there are a few additional GSH commands that need to be run. To do this, run: gsh ../misc /postGrouper2_2Upgrade.gsh (The gsh script is in the "misc" directory.) Note you should check the output to make sure no errors are thrown. If you see an error, it is safe to re-run. For instance..

```
$ ./bin/gsh.sh misc/postGrouper2_2Upgrade.gsh
Using GROUPER_HOME: /opt/grouper
Using GROUPER_CONF: /opt/grouper/conf
Using JAVA: java
using MEMORY: 64m-2000m
Grouper starting up: version: 2.2.0, build date: 2014/06/03 14:30:19, env: <no label configured>
grouper.properties read from: /opt/grouper/conf/grouper.properties
Grouper current directory is: /opt/grouper
log4j.properties read from:   /opt/grouper/conf/log4j.properties
Grouper is logging to file:   /opt/grouper/logs/grouper_error.log, at min level WARN for package: edu.
internet2.middleware.grouper, based on log4j.properties
grouper.hibernate.properties: /opt/grouper/conf/grouper.hibernate.properties
grouper.hibernate.properties: sa@jdbc:hsqldb:hsql://localhost:9001/grouper
sources.xml read from:        /opt/grouper/conf/sources.xml
sources.xml groupersource id: g:gsa
sources.xml groupersource id: grouperEntities
sources.xml jdbc source id:   jdbc: GrouperJdbcConnectionProvider
Type help() for instructions


##########################################
# Grouper 2.2 Upgrade Step 1/7: Clear the temp change log before we start
##########################################
loader ran successfully: Ran the changeLogTempToChangeLog daemon


##########################################
# Grouper 2.2 Upgrade Step 2/7: Add group sets for new privileges (groupAttrRead, groupAttrUpdate,
stemAttrRead, etc)
##########################################


Searching for all composite groups to cache for later use
Searching for missing self groupSets for groups
Found 20 missing groupSets
Done making 20 updates


Searching for missing self groupSets for stems
Found 18 missing groupSets
Done making 18 updates


Searching for missing self groupSets for attribute defs
Found 24 missing groupSets
Done making 24 updates


##########################################
# Grouper 2.2 Upgrade Step 3/7: Add new privileges to point in time (groupAttrRead, groupAttrUpdate,
stemAttrRead, etc)
##########################################


Searching for missing active point in time fields
Found 6 missing active point in time fields
Done making 6 updates
java.lang.Long: 6


##########################################
# Grouper 2.2 Upgrade Step 4/7: Remove old fields (legacy attributes) from point in time.
##########################################


Searching for point in time fields that should be inactive
```

```
Found 16 active point in time fields that should be inactive
Done making 16 updates
java.lang.Long: 16


##########################################
# Grouper 2.2 Upgrade Step 5/7: Migrate legacy attributes
##########################################


Searching for legacy group types
Found 7 legacy group types
Done making 7 updates.  Number previously migrated = 0.


Searching for legacy attributes
Found 16 legacy attributes
Done making 16 updates.  Number previously migrated = 0.


Searching for lists
Found 3 lists
Done making 3 updates.  Number previously migrated = 0.


Searching for group type assignments
Found 5 group type assignments
Done making 5 updates.  Number previously migrated = 0.


Searching for attribute assignments
Found 3 attribute assignments
Done making 3 updates.  Number previously migrated = 0.
java.lang.Long: 34


##########################################
# Grouper 2.2 Upgrade Step 6/7: Add stem sets
##########################################
Searching for missing self stemSets
Found 9 missing self stemSets
Done making 9 updates


Searching for all stems to find existing stem sets with issues
Found 12 stems
Done making 3 updates
java.lang.Long: 12


##########################################
# Grouper 2.2 Upgrade Step 7/7: Add group sets for new privileges to point in time (groupAttrRead,
groupAttrUpdate, stemAttrRead, etc)
##########################################


Searching for missing active point in time group sets
Found 64 missing active point in time group sets
Done making 64 updates
java.lang.Long: 64
```

Note that for the 7th step above (Add group sets for new privileges to point in time (groupAttrRead, groupAttrUpdate, stemAttrRead, etc)), if it will be adding more than 100,000 group sets, then it will process them 100,000 at a time. If you would like to see the progress as it is processing each set of 100K, tail the grouper_error.log file.

10. Run: gsh ../misc/postGrouper2_2_1Upgrade.gsh (The gsh script is in the "misc" directory.)

```
##########################################
# Grouper 2.2.1 Upgrade Step 1/1: Remove MEMBER, ADMIN, UPDATE, and GROUP_ATTR_UPDATE privileges
assigned to GrouperAll
##########################################
Removing GrouperAll membership: groupId=6aac0469f7f24f47ba465ca4b14e8525, group name=etc:test1,
field=admins.
Removing GrouperAll membership: groupId=6aac0469f7f24f47ba465ca4b14e8525, group name=etc:test1,
field=updaters.
Removing GrouperAll membership: groupId=5d3562fb1f6d4c0daebc5b9bf9dce78c, group name=etc:test2,
field=members.
Finished running successfully.  3 changes made.
```

11. Analyze your tables. (Again to avoid any performance issues.)
12. If you are upgrading from 1.6 or 2.0, take a look at item #16 in the 2.1 upgrade instructions.
13. Start the Grouper Daemon and all other Grouper components (UI/WS).
14. The legacy attributes were backed up in separate tables. After verifying that everything is okay, you can drop those backed up tables by setting ddlutils.dropLegacyAttributes = true in grouper.properties, running gsh -registry -deep and then executing the SQL file.

See Also

Grouper Changes v2.2

Release Notes for Grouper 2.2

Upgrade Instructions