foxtechnologies.

BoKS Manager 7.0

Unix Groups Migration Guide

Contact Information

See the Fox Technologies, Inc. Web site for contact information:

Fox Technologies, Inc: www.foxt.com

Trademarks

BoKS, ServerControl, and WorkstationControl are registered trademarks and BoKS Desktop, BoKS Agent, ApplicationControl Suite, AccessControl Suite, AccessControl for Applications, CorporateControls, MailReport, and the FoxT logo are trademarks of Fox Technologies, Inc. or its subsidiaries.

Open Source Software

Some of Fox Technologies, Inc. products include open source software. Copyright and licensing information for such software is provided in the product distribution package.

Third Party Products

Modules included in Fox Technologies, Inc. software may include third-party software. Other product and company names mentioned herein may be the trademarks of their respective owners.

License agreement

This software and the associated documentation are proprietary to Fox Technologies, Inc., are furnished under license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright below. This software and any copies thereof may not be provided or otherwise made available to any other person.

First published: 03/23/2016, Date of this revision: 03/23/2016

Contents

Preface v
Reference Materials v
Getting Support and Service vi
BoKS Unix Groups Migration Guide 7
Purpose and Scope 7
Background and Preparation 7
Background 8
Planning and Preparation 9
The Migration Process 9
High-Level Migration Steps 9
The Migration Tool 10
Migration Report 13
Migration Instructions File - cmds 15
Getting Started with Issue Resolution 16
Migration Script 20
An Example of Migrating Unix Groups 21
Results of the Group Analysis 22
gids Mapped to Multiple Groupnames: gid 564 Cleanup 22
Groupnames mapped to multiple gids: groupE Cleanup 23

Preface

BoKS Manager 7.0 Unix Groups Migration Guide describes how to migrate Unix group configurations in your BoKS domain when you upgrade from a pre-7.0 version of BoKS Manager to version 7.0 or later.

Note: See www.foxt.com for updated versions of product documentation.
Revision: 03/23/2016

Audience

This guide is intended for administrators responsible for upgrading and configuring a BoKS domain.

Some specific knowledge, apart from a general computer literacy, may be required of an administrator:

- Understanding BoKS concepts and database objects
- Configuring files on the BoKS Master

How This Guide Is Organized

The Migration Guide is divided into the following sections:

- Purpose and Scope which outlines the aims and limitations of this Migration Guide.
- Background and Preparation which introduces why migration is necessary and how you prepare for the process.
- The Migration Process which goes through the steps required to migrate Unix Groups in your domain.
- An Example of Migrating Unix Groups which guides you through an example migration scenario.

Product Documentation

The BoKS Manager documentation is available from the FoxT Support web site, which can be accessed via the main website at www.foxt.com.

Reference Materials

You can obtain the following primary documentation from your Fox Technologies Representative, from the FoxT online Documentation Library, or in some cases online from the product itself after the product is installed.

FoxT Product Documentation

- BoKS Manager Administration Guide
- BoKS Manager Installation Guide

Getting Support and Service

Fox Technologies, Inc	www.foxt.com

Before You Call for Technical Support

Note: Technical support is not provided during the warranty period unless a valid Software Service Contract is in force.

Please have the following information available when you call:

- Your Fox Technologies Customer / License ID.
- Fox Technologies software version number.
- The make and model number of the computer on which the problem occurs.
- The name and version of the operating system under which the problem occurs.

1 BoKS Unix Groups Migration Guide

The Unix Groups Migration Guide includes the following sections:

- Purpose and Scope
- Background and Preparation
  - Background
  - Planning and Preparation
- The Migration Process
  - High-Level Migration Steps
  - The Migration Tool
  - Migration Report
  - Migration Instructions File - cmds
  - Getting Started with Issue Resolution
  - Migration Script
- An Example of Migrating Unix Groups

Purpose and Scope

The purpose of this Guide is to describe how to migrate the Unix Group configurations in your BoKS domain when you upgrade from a pre-7.0 version of BoKS Manager to version 7.0 or later.

This guide does not attempt to provide an exhaustive description of all possible issues that could arise when migrating Unix Groups, but describes the tools provided by FoxT to aid migration, and is a representation of some common considerations and issues that may be applicable when you upgrade your BoKS domain(s).

Background and Preparation

This section includes the following topics:

- Background
- Planning and Preparation

Background

This section describes the background to the different way Unix Groups are handled in pre-BoKS 7.0 and BoKS 7.0 and later, and why this requires you to migrate groups when upgrading.

Unix Groups Pre-7.0

In pre-7.0 BoKS versions, secondary UNIX groups are simply stored as gid/name pairs. It is allowed to have a gid map to multiple names and a name map to multiple gids.

Groups are not automatically provisioned to hosts / Host Groups. This only happens when a gid is assigned to a user (directly or via a User Class) with secondary Unix Group support turned on. In this case, an update request is sent to all the hosts in the user's Host Group.

Each BoKS Server Agent then updates the group file. If the gid is already present, the user is simply added to it without checking the name. If the gid is not present, the Server Agent queries servc for a name to use, and servc simply returns the first available name for that gid.

For more information on configuring and working with Unix Groups in BoKS Manager versions earlier than 7.0, see the *BoKS Manager Administration Guide* for the version you are using.

Unix Groups 7.0 and Later

In BoKS 7.0 and later, secondary Unix Groups are stored as `host(group):groupname,gid`. For all hosts in the system of type `UNIXBOKSHOST`, a gid (in BoKS) must map uniquely to a groupname, and a groupname (in BoKS) must map uniquely to a gid.

If there are conflicts, it is not possible to add the Unix Group. When Unix Groups are created, they are automatically provisioned as long as hosts are running BoKS 7.0 and later (this feature is not important for the migration tool).

The Unix Group in the format `host(group):groupname` can then be assigned to users or User Classes.

This difference in format and conflict-handling means it is not generally possible to perform an automatic migration of Unix Group data from a pre-7.0 to a 7.0 and later system. Conflicts may arise that cannot be automatically resolved.

Therefore, when you upgrade a pre-7.0 database to 7.0 or later, all Unix Group data is dropped, and secondary Unix Group support turned off for all users. Turning off the Unix Group support for all users means the secondary group membership will be preserved on all Server Agents, so users logging in will see no change. In order to migrate Unix Group data, you need to use a tool that reads the pre-7.0 database dump and provides output to help you resolve any conflicts.

For more information on configuring and working with Unix Groups in BoKS Manager 7.0, see the chapter *Unix Group* in the *BoKS Manager 7.0 Administration Guide*.

Planning and Preparation

When you are planning an upgrade to BoKS 7.0, and if you are using BoKS for Unix group management, it is advisable to plan the Unix Group migration from an early stage in the upgrade process.

Unix Group migration is an iterative process that may take some time to complete, depending on your environment, and is best started on well before you perform the actual upgrade.

See also “Sequence for Performing the Steps” on page 10.

The Migration Process

This section includes information on how the migration process works and how to use the migration tool to transition Unix Group assignments in your domain to the BoKS 7.0 and later format.

It includes the following topics:

- High-Level Migration Steps
- The Migration Tool
- Migration Report
- Migration Instructions File - cmds
- Getting Started with Issue Resolution
- Migration Script

High-Level Migration Steps

The high-level process to perform a migration is as follows:

1. Import a pre-7.0 database to the Migration Tool with the `migrate.pl -i` option (this saves data to a separate file for quicker execution).

   The database must be in the output format produced by running **dumpbase**, not the format produced by **boks_bru**. For details see “To run the Migration Tool:” on page 13.

2. Produce a report on the database using the Migration Tool.
3. Update the instructions file based on the report.
4. Repeat steps 2 and 3 until satisfied.
5. Create a migration script based on the finalized instructions file.
6. Execute the migration script on the 7.0 Master.

Sequence for Performing the Steps

It will probably take time to perform the steps in the migration process, so you want to start before doing the actual upgrade to 7.0.

The way to solve this is to take a database dump well before the upgrade is done and perform the steps 1 - 4.

Just before the upgrade, take a new database dump, import it (using `migrate.pl -i <databasedump>`) and re-run steps 2 - 4 if needed.

Before upgrading the Master, make a database backup with **dumpbase** as well as using **boks_bru** on the pre-7.0 Master. The first will be used for the Migration Tool, the second to restore the database on the 7.0 Master.

Finally, after the upgrade, take the **dumpbase**-format database dump, import it (using `migrate.pl -i <databasedump>`) and do steps 2 - 6.

The last run through of steps 2 - 6 is designed to make sure there are no last minute changes to the database that need to be processed. Hopefully only minor adjustments are needed after the initial instructions file is generated.

For information on performing an upgrade, see the *BoKS Manager Installation Guide* for your BoKS Manager version.

The Migration Tool

The BoKS Unix Group Migration Tool is delivered as a tar archive to be unpacked in an empty directory.

The Migration Tool can be run on any Unix/Linux host, with the following requirements:

- Perl must be installed. Tested versions of Perl are the BoKS Perl binary (**pcb**) or Perl version 5.14 or later
- Sufficient RAM to execute. For example with a large BoKS database, the tool may require up to half a Gigabyte or RAM to execute.

The tool consists of the following components when unpacked:

- the Perl program **migrate.pl**
- **stylesheet** sub-directory containing support files for the html reports
- **cmds**, a template instruction file for the **migrate.pl** program

Once you have run the **migrate.pl** program, further files and subdirectories are created:

- **data**, a file containing only the data required for the migration from the database dump
- **index.html**. the homepage of the html report when a report has been created
- **html** sub-directory, potentially containing a large number of html files used in the reports

How the Migration Tool Operates

The Migration Tool only concerns itself with users that have the secondary Unix Group support flag set (except for the report on users with gids assigned but flag not set) and hosts of type `UNIXBOKSHOST`. All references to users and hosts in this document are to these types of users / hosts. This means, for example, that you cannot use the `-U` option in the tool (see "Migration Tool Usage" on page 12) for a user without the Unix Group flag set to report on changed gids, as the tool will then return the error "No such user".

The Migration Tool reads the BoKS database dump and for each gid in the database, it finds all users with secondary Unix Group support turned on that the gid is assigned to. It then attempts to internally auto-generate all Unix Groups of the form `host(group):groupname(gid)` needed and assign them to users and User Classes to ensure that in the resulting 7.0 database, all hosts and users will have the same gids and groupnames as before. It then checks for conflicts.

With one exception, the tool will ignore any inconsistencies in the database. For example, if there is a mapping from a gid to a user and that user does not exist, that user is ignored. The one exception is if a gid is mapped to a user or User Class, and that gid has no name, this is reported as an issue, and it is possible to fix this.

The tool can create a report detailing the findings from the scan of the database (see "Migration Report" on page 13).

For example, if the old database has a group `ugroup` with gid `4711`, the user `HGRP1:user1` has the gid `4711` assigned and the Unix Group flag turned on, and the user `HGRP2:user2` with the Unix Group flag turned on is a member of the User Class `CLASS` which has the gid `4711` assigned, the tool will internally generate two Unix Groups, `HGRP1:ugroup(4711)` and `HGRP2:ugroup(4711)`. The first one will be assigned to the user `HGRP1:user1` and the second one to the User Class `CLASS`.

There are three types of issues that can arise when the tool is run that require manual intervention:

1. A gid in the original database maps to multiple group names. In this case the tool does not know what name to choose, and you must do this manually (in fact you may want different names for different host(groups)).

2. A group name maps to multiple gids. This may lead to a conflict where the tool attempts to create e.g. `HGRP1:group1,gid1` and `HGRP2:group1,gid2`. This will lead to a conflict if there is one or more `UNIXBOKSHOST`(s) that exist in both Host Groups (group name does not map uniquely to a gid). This must be resolved manually.

3. The tool may create a large number of `host(group):groupname,gid` objects for each gid. It may be possible to instead create fewer of these by assigning them to larger Host Groups. This consolidation must be done manually.

In the first two cases, the migration tool will simply not create the groups that would cause problems. This will also lead to the tool reporting that users and hosts are missing gids.

These issues can be resolved by creating an instructions file that is read after the database dump and updates the internal Unix group configuration according to the instructions you provide. See “Migration Instructions File - cmds” on page 15.

Migration Tool Usage

The migration tool program **migrate.pl** has the following usage and options:

Option	Description
-i dbdumpfile	Initialize from the database dump file dbdumpfile. This must be done any time the database dump is changed.
-r	Generate an html report. To view the report, point your browser to the index.html file generated in the same directory as the migration tool.
-t	Generate a text report on stdout.
-O	Generate an html report on overlapping Host Groups. This is a separate step as it may take a long time, and may not be needed. The report must be re-generated if the Host Group configuration is changed by commands in the cmds file (specifically the add_members command).
-T	Generate a text report on overlapping Host Groups. Please see option -O for additional information on this report.
-o host	Show Host Groups that the host `host` belongs to on stdout.
-o hostgroup1 hostgroup2	Show overlapping hosts between the specified Host Groups on stdout. Performing this step may be quicker than consulting the overlapping Host Group report. To list all the hosts in a Host Group, type `-o hostgroup ALL`.
-h host / hostgroup	Show users for `host` / `hostgroup` on stdout.
-U user ...	Show changed gid for user(s) on stdout.
-u user ...	Show gids assigned to users in the original database (either directly or via User Class) on stdout.
-c userclass	Show all users in `userclass` on stdout.
-M min	Print out gids with at least `Min` number of BoKS 7.0 (or later) Unix groups.
-m gid	Print out the hosts / Host Groups for the Unix groups for which `gid` is created.
-g	Generate shell script on stdout.
-G	Generate only shell comments on actions to perform.
-H	Display migration tool help.

Running the tool

To run the Migration Tool:

1. Log in to the machine where the tool is installed.
2. Move to the directory where the tool is located.
3. Run the **migrate.pl** program with the required options.
   For example, to scan the database dump located in **/tmp/boksdb**:

   ```
   # migrate.pl -i /tmp/boksdb
   ```

   To show which Host Groups the host `host1` belongs to on stdout:

   ```
   # migrate.pl -o host1
   ```

   To show overlapping hosts between the Host Groups `HG1` and `HG2`:

   ```
   # migrate.pl -O HG1 HG2
   ```

Migration Report

After initializing data, the tool can analyze Unix Group assignment data in the database and text or html reports can be generated. The migration report is produced by running the tool **migrate.pl** with the option `-r` (for an html report) or `-t` (for a text report) on the BoKS database dump. A text report (`-t`) is written directly to stdout and can be very long, but may be useful for reference or as a source for scripting special things.

An html report (`-r`) can be more useful in that it can be navigated. Just point you browser to the **index.html** file generated. The html reports work with Internet Explorer 9 or later, and have alse been tested with Microsoft Edge and recent versions of Chrome and Firefox.

Note: It is **not** supported to run the html reports on Internet Explorer 8 or earlier.

Note that with a large database, it can take a few minutes to generate a report.

The report includes information about:

- What operations the migration tool would actually perform if run on the database (with the exception of created Host Groups and members added to Host Groups).
- Any Unix Group conflicts and multiple group name problems.
- Differences in gid assignments to users from the original data.
- Differences in gid provisioning to individual hosts from original data.

When you have created a report, you can use the information it contains to update the **cmds** instructions file for the migration program to remove any conflicts, multiple name problems, and differences for users and hosts.

Once you have created an instructions file that does this, the migration tool can be run to produce a script that will perform the required operations on a BoKS 7.0 Master with the original database imported.

Creating a Report

Note: Before you can create a report, you must have imported a BoKS database using the `migrate.pl -i <dbdumpfile>` command.

To create a report using the Migration Tool:

1. Log in to the machine where the tool is installed.
2. Move to the directory where the tool is located.
3. Run one or both of the following command(s):

   ```
   # migrate.pl -t
   ```

   To create a text report to stdout.

   ```
   # migrate.pl -r
   ```

   To create a html report that can be accessed via the **index.html** file.

Report Contents

The report includes multiple sections with information described in the following table:

Section / Subsection	Description
Issues section	
Gids with multiple names assigned to users or userclasses (will not be created)	Gids that map to multiple names. As the tool does not know what name to use where, it does not automatically generate any Unix Groups for these gids.
Gids with no name assigned to user or userclass (ignored)	This is only displayed if there is an inconsistency in the database such that a user or User Class has a gid assigned, but the gid has no name. This will also show up as hosts and users with missing gids.
Overlapping unixgroups (same gid different names) (will not be created)	This will only occur as the result of manual configuration in the instructions file, cmds. A manually created Unix Group has an overlap with an auto-generated Unix Group or another manually created Unix Group and they have the same gid but different group names. Neither the Unix Groups in the cmds file nor the auto-generated groups will be created.
Overlapping unixgroups (same name different gid) (will not be created)	These are overlapping auto-generated or manually created Unix Groups where a group name maps to multiple gids on some hosts. Neither the Unix Groups in the cmds file nor the auto-generated groups will be created.
Hosts with changed gids	These are hosts that would get a different set of gids compared with the old database. This can be because of groups that could not be generated because of the three cases described above, or because Unix Groups were manually added or removed. This may or may not be acceptable.

Section / Subsection	Description
Users with changed gids	These are users that would get a different set of gids compared with the old database on some or all hosts. This can be because of groups that could not be generated due to the three cases described above, or because Unix Groups were manually added or removed. This may or may not be acceptable. If a Unix Group is manually assigned to a user, but there is no overlap between the host / Host Group for the Unix Group and the host / Host Group for the user, this is reported as "Added gids (but not in users host / Host Group)". The group will still be assigned to the user.
Actions section	
Unixgroups to create (per gid)	This displays Unix Groups per gid that would be created with the current configuration, and the users and User Classes they would be assigned to.
Information section	
Users with groups assigned but secondary unix group flag not set	This lists users in the old database that have gids directly assigned but do not have the unix group flag set. The tool will ignore these users.
Overlapping hostgroups	This section is only included in the html report if the tool is run with the -O flag first. It displays overlapping Host Groups, and may be useful if you want to change the way Unix Groups are created to Host Groups. Generating this report can take several minutes for a large database. If Host Groups are modified in any way in the cmds instruction file, the report must be manually regenerated to be up to date.
gids not assigned to any user or userclass (will not be created)	These are gids in the old database that are not assigned to any user or User Class, or are assigned (directly or via User Class) to users without the Unix Group flag set. As the tool does not have any way of knowing what hosts or Host Groups to create a Unix Group to, it will not create these Unix Groups.

Migration Instructions File - cmds

The migration instructions file (a text file named **cmds**) is used by the migration tool to produce a migration script. The script is then used on the upgraded BoKS Master to update the database with new Unix Groups and assignments.

A template **cmds** instruction file that lists the supported commands is included in the Migration Tool package. Empty lines in the file are ignored, and you can include comments by starting the line with #.

Note that any time you modify Host Groups using the **cmds** file, you must manually regenerate the migration report (either text or html version) to ensure that the report is up to date.

The **cmds** file can include the following instructions:

Instruction	Description
ignore_gids gid [gid...]	Instruct the tool to completely ignore the specified gids. Do not create Unix Groups for them, and do not report any issues for them.
delete_groups host(group):groupname ...	Instruct the tool not to automatically create the specified Unix Groups.
delete_gids gid ...	Instruct the tool not to generate any Unix Groups for the given gid(s).
create_group host(group):groupname gid	Manually create a Unix Group.
assign_group2users host(group):groupname user ...	Assign a group created with create_group to the listed users.
assign_group2users_via_hgs host(group):groupname host(group) ...	Assign a group created with create_group to all users in the Host Groups listed that have the corresponding gid assigned in the old database. If host (group) is * the Unix Group will be assigned to all users with the corresponding gid assigned.
assign_group2userclasses host(group):groupname userclass ...	Assign a group created with create_group to the listed User Classes.
assign_group2userclasses_via_hgs host(group):groupname host(group) ...	Assign a group created with create_group to all User Classes that have the corresponding gid assigned, and at least one user (with Unix Group flag set) in one of the listed Host Groups (again, * matches any Host Group).
create_hostgroup hostgroup	Instruct the tool to create a Host Group.
add_members hostgroup host ...	Instruct the tool to add the specified hosts to the specified Host Group. Hosts must be the names of UNIXBOKSHOSTs in the old database and cannot contain wildcards.

Editing the Instructions File

You can simply update the **cmds** file in any file editor to configure the instructions you want to provide to the **migrate.pl** script. The script reads this file any time it is run, as long as it is in the current directory where you are running **migrate.pl**.

Getting Started with Issue Resolution

First generate a report just from the database dump. There will probably be gids with multiple names or conflicts. Start by attempting to resolve these one by one.

The first step is to add "`ignore_gids`" in the instructions file for all the gids in all these issues. For gids with multiple names, put each one on a separate line, but for the overlapping issues, put all gids for the same reported issue on the same line.

Now comment out the first of these and generate a new report. Now only this issue will be reported, making it much easier to sort out just that issue. After commenting out the `"ignore_gids"` command for that gid, add commands to attempt to resolve the issue (see suggestions below) and then generate a new report to see if the issue is fixed.

The following sections describe how specific types of issues can be handled with examples:

- Gids with multiple names
- Gids Assigned to User or User Class with no Name
- Overlapping Unix Groups (Same gid Different Names)
- Overlapping Unix Groups (Same Name Different gids)
- Hosts and Users with Changed gids
- Unix Group Consolidation
- Final Steps

Gids with multiple names

In this case, look at the Host Groups for which the gid should have been added. In the instructions file you can now do "`create_group`" for each Host Group with the group name you want and assign the created groups to the appropriate users and User Classes. Before doing this you may want to check that the Host Groups do not overlap (using the overlap report or running the tool with `-o hgrp1 hgrp2`).

Issue Resolution Example - gids with multiple names

Users `H1:user1` and `H2:user2`, both with gid `4711` assigned.

Gid `4711` is mapped to names `group1` and `group2`, so the initial report will complain that the gid has multiple names. To resolve that, add these lines to the **cmds** file:

```
# Assume we want gid 4711 to have the name group1 for hosts in H1
create_group H1:group1 4711
# Assign to all users in H1 that have this gid
assign_group2users_via_hgs H1:group1 H1
# If gid is assigned to userclasses, you probably also want
# assign_group2userclasses_via_hgs H1:group1 H1
#
# And we use the name group2 in H2
create_group H2:group2 4711
# If you want full control over what users get the unixgroup:
assign_group2users H2:group2 H2:user2
```

Now run the tool again to check the result. If there was an overlap between the Host Groups, you would get a new issue: Overlapping Unix Groups (same gid different names). Resolving this is more complicated - for more information, see “Overlapping Unix Groups (Same gid Different Names)” on page 18.

Gids Assigned to User or User Class with no Name

This can be solved by simply using `create_group` to create the required Unix Groups with the name they should have and then assigning these to the correct users and User Classes. If the gid is no longer in use, simply keep the `ignore_gids` statement so it is not reported as an issue. No groups will be created for the gid.

Overlapping Unix Groups (Same gid Different Names)

As this is a result of one or more Unix Groups created in the **cmds** file, you must try to come up with another approach when creating the Unix Group(s). Either by creating new Host Groups to assign the Unix Groups to in order to avoid the overlap, or by changing names or gids.

Here is an example of how this can happen:

You have two users `H1:user1` and `H2:user2` where the Host Groups overlap.

If you now add commands in the **cmds** file to create `H1:group1` with gid `4711` and assign it to user `H1:user1` and also create `H2:group2` with the same gid and assign to `H2:user2`, you will get this conflict.

The same thing would happen if `H1:group1` with gid `4711` was auto-generated because gid `4711` with name `group1` existed in the old database and was assigned to `H1:user1` so the tool would auto-generate `H1:group1` and you then create `H2:group2` with gid `4711` and assign to user `H2:user2` in the **cmds** file.

If you want to take full control over how Unix Groups for a given gid are generated, you can stop the tool from auto-generating Unix Groups for this gid by adding:

```
delete_gids 4711
```

Example with auto-generated Unix Groups

Let's say you have three hosts `h1` (in `H1`), `h2` (in `H2`) and `overlap` (in `H1` and `H2`). Assume you want the gid `4711` to have the name `group2` on host `overlap`. In this case you need to create an extra Host Group with the hosts in `H1` that do not overlap `H2` (only `h1` in this case), and then create Unix Groups appropriately:

```
# Take control over unix group generation for gid 4711:
delete_gids 4711
create_hostgroup H1_NO_OVERLAP
add_members H1_NO_OVERLAP h1
create_group H1_NO_OVERLAP:group1 4711
# Assign to users in H1. As the group does not exist on host overlap,
# there will be no conflict
assign_group2users_via_hgs H1_NO_OVERLAP:group1 H1
#
create_group H2:group2 4711
assign_group2users_via_hgs H2:group2 H2
# and if you want H1 users to be a member of group2 on the overlap
hosts
# (if not the next report will complain that these users are missing
the
# gid on host overlap)
assign_group2users_via_hgs H2:group2 H1
```

The problem with this approach is of course that you get an extra Host Group to maintain. It would be much better if this could be resolved by just deciding to use one of the names, or by changing the gid for one of the names to avoid a conflict that way.

Overlapping Unix Groups (Same Name Different gids)

This can happen as a result of auto-generated or manually created Unix Groups and must be resolved so the name only maps to one gid for the overlapping hosts in order for the Unix Groups to be created.

Example with auto-generated Unix Groups

Hostgroups `H1` (hosts `h1` and `overlap`) and `H2` (hosts `h2` and `overlap`).

In the old database gid `4711` is mapped to `group1` and `4712` is also mapped to that name.

Users `H1:user1` has gid `4711` and `H2:user2` has gid `4712`.

The auto-generated Unix Groups `H1:group1`, gid `4711`, and `H2:group1`, gid `4712`, will overlap on host `overlap`, and there the name `group1` maps to two gids which is not allowed by BoKS 7.0 and later.

You need to take control over how Unix Groups are generated to stop this conflict. For instance to stop `H1:group1` from being auto-generated, add `delete_groups H1:group1`.

You could then for example create a new hostgroup `H1_NO_OVERLAP` as above, create the Unix Group `H1_NO_OVERLAP:group1` with gid `4711` and assign to user `H1:user1`. A new report will now report the issues that gid `4711` is missing on host `overlap`, and that user `H1:user1` is missing that gid on the host. This may or may not be acceptable.

Another way would to change the group name for one of them. For example,

```
delete_group H1:group1
create_group H1:newname 1
assign_group2users H1:newname H1:user1
```

Now there will be no conflict, no missing gids, but the name of the group `group1` is changed to `newname`. This may or may not be acceptable.

Hosts and Users with Changed gids

This is a result of Unix Groups not being created because of issues, or because of commands in the **cmds** file. Of course you want to avoid this, but in some cases it may be desirable. In BoKS 7.0 and later, you have greater control over where Unix Groups are created; A user may not need a secondary Unix Group on all hosts the account exists on, and a Unix Group may not need to exist on some hosts.

Unix Group Consolidation

By itself, the tool may generate a large number of Unix Groups for certain gids (if they are assigned to users in many different Host Groups directly or via User Classes). It may be possible to create fewer Unix Groups to larger Host Groups to simplify administration.

To check for this, use the `-M min` option to the tool. This will list gids (manually or auto-generated) that have at least `min` Unix Groups created to it. You can then consult the **Unixgroups to create** section in the full report to find the Unix Groups that would be created for that gid (or just run the tool with `-m gid` for the gid in question) and try to determine if it would be possible to delete some of the Unix Groups and create a Unix Group to a larger Host Group instead. If you make a mistake, it will show up in the next report generated as hosts and users with changed gids.

Final Steps

Once you are satisfied with the result, you can use the `-g` option to the Migration Tool to generate a script on stdout (redirect it to a file) that can be run from a BoKS prompt on the BoKS 7.0 or later Master to set up the Unix Groups and secondary group membership as reported. When running the script, it will report actions to stdout, and any errors or information messages to stderr.

For more details see the following section, *Migration Script*.

Migration Script

Once you have resolved any conflicts found in the migration report and are happy with the new Unix groups and assignments that will be created, you can generate a migration script that you will use to update the database on your BoKS 7.0 Master.

You can view the operations the migration script will perform before you create it to check that they are correct. The script contains a lot of extra error checking code which makes it hard to see exactly what it will perform, so you can use the `migrate.pl -G` switch to only generate comments on what will be performed.

Important: The generated script will turn on the Unix Group management flag for all users who had it in the old database even if they have no secondary Unix Groups assigned (directly or via User Class).

Once the script is complete it should be run from a BoKS shell on the BoKS 7.0 or later Master to update the BoKS database. The script reports information and any errors to stderr and information about the operations performed to stdout. If run with the option `-t` (test-only mode), the script does not perform any operations, it only reports on actual actions that would be performed to stdout and any information and errors stderr.

The script is designed to be run multiple times if required, detecting any action that has already been performed and reporting this to stderr as `INFO`.

Note: The test-only mode cannot detect all errors. For instance, if a host or Host Group to which a Unix Group should be assigned is missing, it will report an error when the Unix Group should be created, but will still later report that the Unix Group is successfully added to User Classes and/or users.

To view operations the migration script will perform:

1. Run the Migration Tool with the `-G` option:

   ```
   # migrate.pl -G
   ```

   The operations for the migration script are displayed.

To create the migration script:

1. Run the Migration Tool with the `-g` option and send it to a file.
   For example:

   ```
   # migrate.pl -g > migrate.sh
   ```

   To send the script to the shell script file **migrate.sh**.

To run the migration script:

Note: If the generated script is large, it is likely to take a considerable amount of time to run.

1. Save the migration script on the BoKS Master.
2. Log in to the BoKS Master and become a user with administrative privileges.
3. Start a BoKS shell:

   ```
   # /opt/boksm/sbin/boksadm
   ```

4. Run the migration script.
   For example:

   ```
   BoKS # sh /tmp/migrate.sh
   ```

An Example of Migrating Unix Groups

This example scenario is designed to illustrate how an organization upgrading to BoKS Manager 7.0 uses the migration tool to discover possible conflicts in Unix Group assignments and takes action to mitigate these in the new environment.

It includes the following topics:

- Results of the Group Analysis
- gids Mapped to Multiple Groupnames: gid 564 Cleanup
- Groupnames mapped to multiple gids: groupE Cleanup

Results of the Group Analysis

The organization uses **dumpbase** with output re-directed to a file to create a dump of the pre-7.0 BoKS database and runs the Migration Tool on the resulting file.

The resulting report shows that in the domain:

3 gids are mapped to multiple groupnames:

- `564` maps to `groupA`, `groupB` and `groupC`
- `589` maps to `group1` and `group2`
- `986` maps to `group3` and `group4`

3 groupnames are mapped to multiple gids and exist on hosts with overlapping Host Group memberships:

- `groupA` maps to `564` and `554`
- `groupD` maps to `589` and `665`
- `groupE` maps to `342` and `377`

gids Mapped to Multiple Groupnames: gid 564 Cleanup

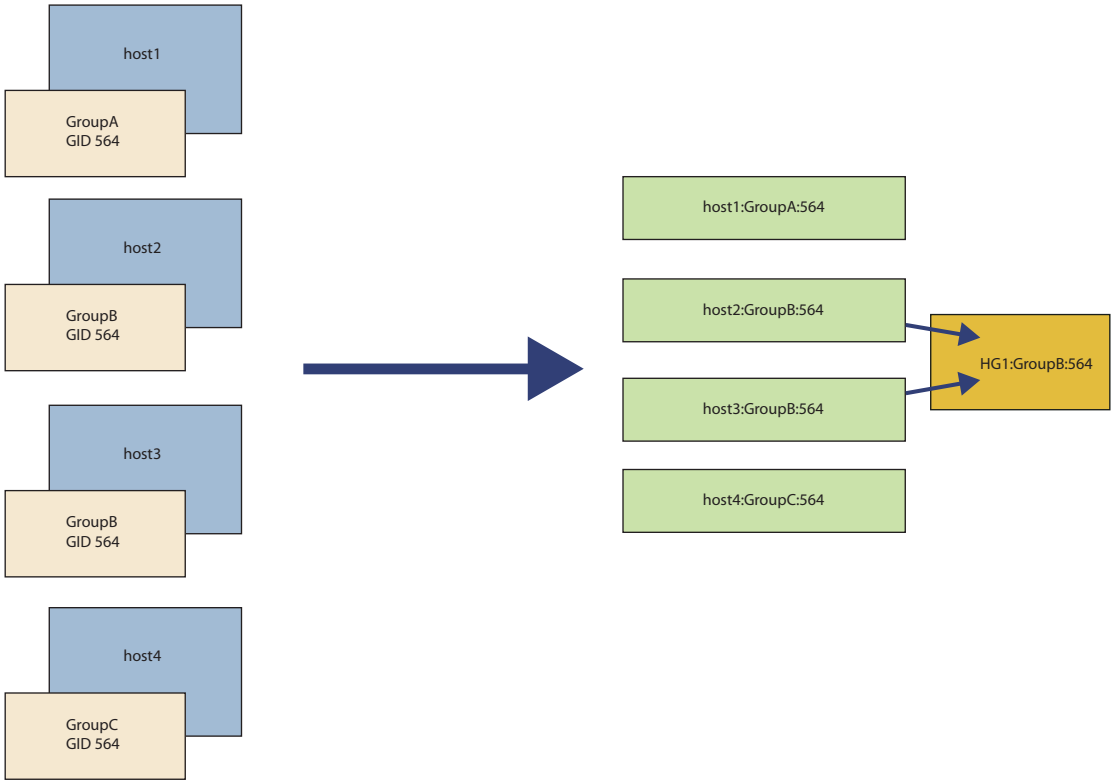
Analysis of gid `564` shows that it is assigned to `groupA` on the host `host1`, `groupB` on the hosts `host2` and `host3` and `groupC` on the host `host4`.

The steps to clean up this situation and create Unix Groups for the BoKS 7.0 domain are as follows:

1. Remove gid `564`.
2. Create the following Unix Groups in the new BoKS format:
   - `host1:groupA:564`
   - `host2:groupB:564`
   - `host3:groupB:564`
   - `host4:groupC:564`

   Note that if appropriate you can also add `host2` and `host3` to the same Host Group, for example `HG1`, then create the Unix Group `HG1:groupB:564`.

3. Assign users and User Classes to the new Unix Groups in BoKS as needed.



Groupnames mapped to multiple gids: groupE Cleanup

Analysis of `groupE` shows that in the old database it maps to gids `342` and gid `377`.

User `H1:user1` has gid `342` and user `H2:user2` has gid `377`. However the host `host5` is a member of the Host Groups `H1` and `H2`.

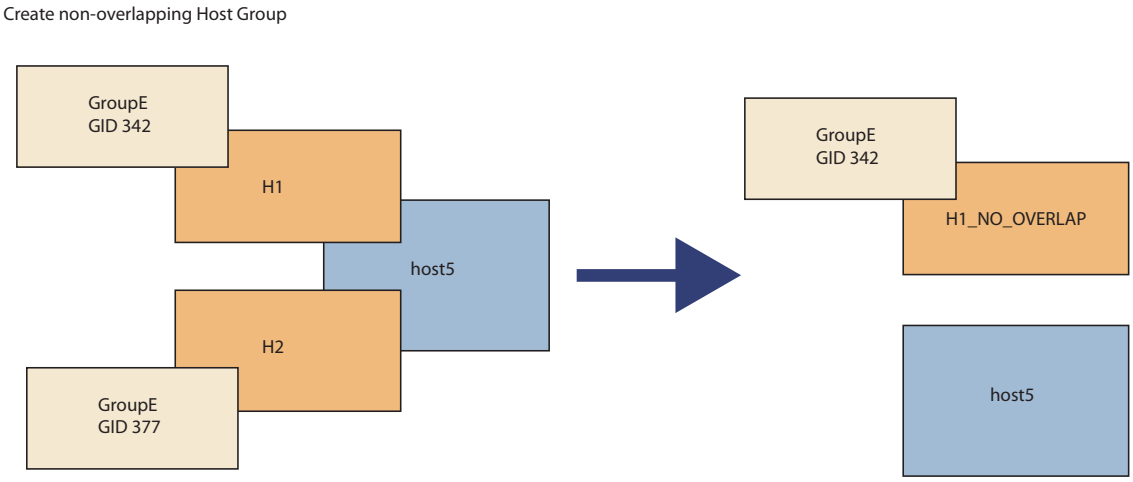
The auto-generated Unix Groups `H1:groupE, gid 342`, and `H2:groupE, gid 377`, will overlap on `host5`, which is a member of `H1` and `H2`, and there the name `groupE` maps to two gids which is not allowed by BoKS 7.0 and later.

There are two possible approaches to cleaning up this situation and creating Unix Groups for the BoKS 7.0 domain:

1. Remove `groupE`.
2. Either:
   - Create the Host Group `H1_NO_OVERLAP` where other hosts are members but not `host5`.

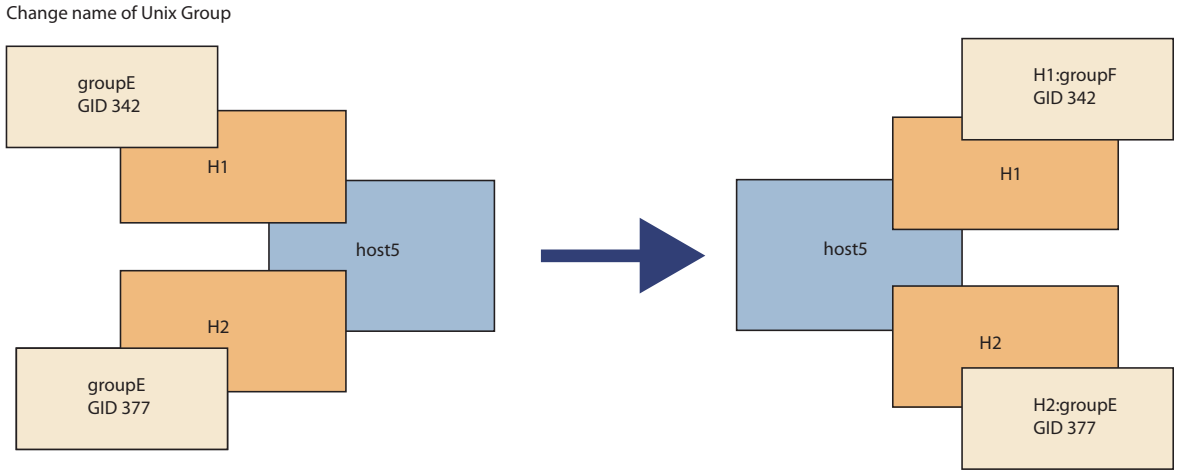
     Create the Unix Group `H1_NO_OVERLAP:groupE` with gid `342` and assign it to the user `H1:user1`.

The tool will report that gid `342` is missing on `host5` and that user `H1:user1` is missing the gid on that host.



- Change the name of `groupE` on one of the Host Groups.
  For example in the instructions file:

```
delete_group H1:groupE
create_group H1:groupF 342
assign_group2users H1:groupF H1:user1
```



Index

C

cmds file
    instructions file for migration 15

E

example
    scenario for migrating groups 21

I

instructions file 15
    using for issue resolution 16
issue resolution
    getting started with 16

M

manuals, reference v
Master
    running the migration script on 21
migration instructions file 15
migration script 20
    creating 21
    running on the Master 21
    viewing operations with migration tool 21
migration steps, high-level 9
migration tool
    description 10

P

planning 9

R

reports
    for migration 13

U

Unix groups
    BoKS 7.0 and later format 8
    pre-BoKS 7.0 format 8