NETAPP GLOBAL SERVICES

SnapVault Automation

Version 0.15
Dec 01, 2010

Prepared by:
Matthew Goldman
Professional Services Engineer
Edina, MN

ABOUT NETAPP

NetApp creates innovative storage and data management solutions that deliver outstanding cost efficiency and accelerate performance breakthroughs. Discover our passion for helping companies around the world go further, faster at www.netapp.com.

NetApp, Inc.
495 East Java Drive
Sunnyvale, CA 94089 USA
Telephone: +1 (408) 822-6000
Fax: +1 (408) 822-4501
Support telephone: +1 (888) 4-NETAPP

Copyright and Trademark Information

Table of Contents

Preface

This document is intended to provide operating instructions for the current process of moving SnapVault backup destination volumes as well as how ongoing cleanup work can be transitioned between different groups working different shifts.

AUDIENCE

The primary audience for this document is PSE Onsite personnel for Thomson Reuters Professional.

NON-DISCLOSURE REQUIREMENTS

FEEDBACK

We continually try to improve the quality and usefulness of NetApp documentation. If you have any corrections, feedbacks, or requests for additional documentation, send an e-mail message to dl-ngs-ps-eng-tw@netapp.com.

1 Introduction

One of the ongoing tasks for an Onsite at Thomson Reuters Professional (TRP) in Eagan is moving SnapVault backup destination volumes. This task is due to thin provisioning of space in the aggregates. This effort makes it is possible for TRP to easily use a larger faction of the available space.

This document is intended to provide operating instructions for the current process as well as details on how ongoing cleanup work can be transitioned between different groups working different shifts.

1.1 Service requirements

It is important to have a clear understanding of the objectives of the automation project and what parts of the process are not being automated. As an ongoing well developed process, this is no longer in need of a CR to perform the work.. The following shows a high level overview of the tasks to be accomplished on a weekly basis:

1. Add suffix of '_FULL' to aggregates that have more than 70% of their actual usage. This is now fully automated via DFM, and no longer part of the manual process.
2. Remove the '_FULL' suffix for aggregates that are currently marked as full when they drop below 65% actual usage. This is now fully automated via DFM, and no longer part of the manual process.
3. Finalize space freeing efforts that are awaiting finalization.
4. Determine where space should be freed up. Space should be freed on aggregates that are more than 90% full. Existing SnapVault relationships should be maintained. Particular attention needs to be paid to aggregates that are more than 95% full.
5. Start process of freeing up space on aggregates.

2 Task overview

After investigation, a number of helper programs were written for tasks. All helper scripts are stored in the three DFM servers under netapp/scripts/Weekly_Aggr_Status.

The tasks are divided into four distinct phases - information gathering, initiating SnapMirror transfers, awaiting the completion of the initial SnapMirror baseline transfer, and finalizing SnapMirror transfers. One or more helper scripts have been written for each phase.

One of the helper scripts is automatically created. This is discussed in detail below. The following is a list of helper scripts:

- **fullness.ksh** – This script gives a quick overview of the current state of aggregates on all of the backup storage controllers. The list is sorted by percent used (low to high) and has breaks at various percentages of interest. It is intended to be run during the first phase; however, it is useful during all phases.
- **Run_them_all.ksh** – This script has been superseded by 0_check_status.pl.
- **check_outstanding_jobs** – This script has content automatically added to it. It is useful during all phases because it keeps track of outstanding work. In the early phases, it is useful to prevent the operator from attempting to move one aggregate twice. In the later phases, it helps you keep track of what active jobs are ready for finalization. Lines are only added to this file. It is necessary to manually delete old status checks when the data migration work is complete.
- **0_check_status.pl** – This script will generate the commands necessary to start an aggregate migration. This script requires a manually generated configuration file. Details are discussed in later sections.
- **1_start_move.ksh** – This script will generate the commands necessary to start an aggregate migration. This script requires a manually generated configuration file. Details are discussed in later sections.
- **2_finalize_move.ksh** – This script generates some automatic scripts that can be run after validation, and a number of manual scripts that require operator interaction. It uses the same configuration file format as 1_start_move.ksh. It is discussed in detail in later sections.

It is important to note that most of the scripts can be run from any of the DFM storage controllers; however, it is recommended that you do all work from **nidaros**. There are temporary files that get written and are used for later phases. Consistent use of a single DFM storage controller will prevent confusion if more than one person is involved in this cleanup work.

3 Phase 1: information gathering

This phase involves looking at the status of aggregates being used as SnapVault destinations and determines volumes that need to be moved. There are three tools to help you in this phase:

3.1 fullness.ksh

This script takes no arguments and can be run from any of the DFM servers. The output is identical regardless of which DFM server you run it from. It produces a list of available space on all the backup storage controllers. The list of storage controllers is currently hardcoded in the script; hence as additional backup storage controllers are brought online they will need to be added to this script.

This script is useful to determine which aggregates are full, and which ones have space available.

This script takes no input and has no effect on other files on the dfm server.

3.2 0_check_status.pl

This script takes no arguments and can be run from any of the DFM servers. The output is identical regardless of which DFM server you run it from. It runs a number of commands on each of the backup storage controllers. Its output gives important information:

1. What aggregates are above 90% usage.
2. For aggregates that are above 90% usage, nearly complete configuration lines are returned. Each line is tagged with the actual space consumed in the Aggregate by the volume, and the total configured space.

Generally, this command is run and the output is dumped into a file named with the day of the week and the month, day, year. For example, both tuesday_06_07_2010 and wednesday_5_5_2010 would be good file names. This naming helps keep track of the work done.

It becomes very easy to pick which volumes are good candidates for migration.

This script takes no input and has no effect on other files on the dfm server.

The script output looks as follows:

```
#
# Aggregate aggr20_32_FULL on storage controller eg-nasbkp-f01-mgmt is at 90% utalization.  Space can be cleared as
follows (Used Space / Allocated Space):
#

#       eg-nasbkp-f01-mgmt ecomf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_14_nv_pubrecproddb_snap0 6t --
Savings of 2712GB/6144GB
#       eg-nasbkp-f01-mgmt ecomf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_14_nv_erdnorm6p_s01ora1_snap 9242g --
Savings of 3661GB/9242GB
#       eg-nasbkp-f01-mgmt ecomf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_14_crm_advisor1p_s01ora1_snap 270g --
Savings of 19GB/270GB
#       eg-nasbkp-f01-mgmt corpf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_14_bis_mfg1d_s03ora3db_snap 98g --
Savings of 22GB/98GB
#       eg-nasbkp-f01-mgmt corpf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_14_nv_novusbbnv7c_s01ora1_snap 150g -
- Savings of 8GB/150GB
#       eg-nasbkp-f01-mgmt ecomf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_scholarone_volstage_snap 10205g --
Savings of 685GB/10205GB
#       eg-nasbkp-f01-mgmt corpf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_14_ct_cobalt11p_s01ora1_snap 4000g --
Savings of 2678GB/4000GB
#       eg-nasbkp-f01-mgmt ecomf1 aggr20_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_45_tfprod_snap0 4000g -- Savings of
1845GB/4000GB

#
# Aggregate aggr03_32_FULL on storage controller eg-nasbkp-f01-mgmt is at 96% utalization.  Space can be cleared as
follows (Used Space / Allocated Space):
#

#       eg-nasbkp-f01-mgmt ecomf1 aggr03_32_FULL NEW_FILER NEW_VFILER NEW_AGGR sv_scholarone_vol2_snap 14t -- Savings
of 13331GB/14336GB
```

Note that because of the way that this program gathers and displays its data, all volumes with less than 1GB of allocated space are ignored.

As you can see, the lines are nearly ready to be the input for the other commands. The leading '#<tab>' and everything after the two dashes need to be removed. Additionally, the NEW_FILER, NEW_VFILER and NEW_AGGR fields need to be replaced with valid values.

3.3 check_outstanding_jobs

This script has content added to it automatically during Phase 2. It helps keep track of migrations currently in progress. It is a useful tool to prevent duplication of effort, and to ease turnover between different people monitoring and alleviating space issues.

By using the above three scripts, it is possible to determine what work needs to be started. Since the name of the input file is user supplied in later phases, you can use any name you desire.

It is recommended to use the following consistent naming standard:

```
running.cfg
```

I used to keep each day's configurations in their own file; however, I discovered that fewer mistakes were made by having a single configuration file. It is easy to add the date as a comment in the file to keep track of what has been started when if desired.

Additionally, it is important to know that if "1_start_move.ksh" is executed without any options, the program will generate text as shown below. That text can be used as a header for the configuration file as a comment. The text has information about what is expected in each field of the configuration file and also explains that either spaces or commas can be used as a field separator.

This is intended as an aid to building correct configuration files.

```
#       Usage: 1_start_move.ksh Input_file
#
# File format (Space or comma seperated):
# Cur_Backup_Filer Cur_vFiler Cur_Aggr New_Backup_Filer New_vFiler New_Aggr Volume Vol_Size
```

Notice how the output of 0_check_status.pl is nearly in that format already.

> Tip: When adding the current aggregate's name to the configuration file, don't include the '_FULL' in case the aggregate gets renamed while you are doing work. The current aggregate name is only used to build a space checking function and doesn't have to be precise.
>
> You must use the exact name for the new aggregate, as that is used to create a volume and needs to be precise.

The following command line will create a new configuration file that has the header in it. Any contents in the existing file will be deleted.

```
1_start_move.ksh > tuesday_06_07_2010.cfg
```

From the output of Run_them_all.ksh, you need to determine the following:

1. Current backup storage controller
2. Current vFiler
3. Current aggregate

4. New backup storage controller (even if it is the same as the current backup storage controller)
5. New vFiler
6. New aggregate
7. Volume name
8. Volume size (this number may need to be made 10% or so larger, in case the source volume has grown since the information was recorded to DFM.)

One question that comes up frequently is, "What size volumes should be moved?" The answer to that question depends on what volumes are available in the source aggregate. Ideally, look for candidate volumes in the 300 GB to 1,500 GB range. Volumes of that size will transfer fairly quickly. Note that 'fairly quickly' is still one to two days. Sometimes you have no choice but to move a much larger volume. Larger volumes will take much longer to transfer.

Of additional note are the volumes in the 50 GB to 300 GB range. Those will transfer quickly and allow you to rapidly free up space on extremely full aggregates. Generally the space savings isn't worth the effort for aggregates below 50 GB.

An example of a prepped configuration file is as follows:

```
#       Usage: 1_start_move.ksh Input_file
#
# File format (Space or comma seperated):
#  Cur_Backup_Filer Cur_vFiler Cur_Aggr New_Backup_Filer New_vFiler New_Aggr Volume Vol_Size
eg-nasbkp-h01-mgmt ecomh1 aggr07 eg-nasbkp-f01-mgmt ecomf1 aggr02 sv_14_nv_wlnv12p_s01ora1_snap 5529g
eg-nasbkp-h01-mgmt ecomh1 aggr07 eg-nasbkp-f01-mgmt ecomf1 aggr02 sv_14_at_cobaltinfa3p_s01ora1_snap 1200g
eg-nasbkp-e02-mgmt ecome2 aggr23 eg-nasbkp-e02-mgmt ecome2 aggr35 sv_45_nv_cobaltmetadoc1c_s01ora1_snap 5655g
eg-nasbkp-e02-mgmt ecome2 aggr23 eg-nasbkp-e02-mgmt ecome2 aggr38 sv_07_Infra_pv_mn2shrdea_vms_snap07 12737g
```

Once a completed configuration file is ready, go to the next phase. Again, note how closely the above example matches the output of 0_check_status.pl.

3.4 Possible roadblocks

This section will grow as new issues are identified.

The process of determining which volumes to move has been greatly overhauled. On occasion, the vFiler associated with a volume is confused. In those cases, the vFiler name will be set to 'NONE'.

Note: Beware of smart quotes and smart dashes in the above command line. Do not use cut and paste, as the pasted command will fail!

Learn vi. There are only a few commands that you need to know in vi. It will make your life easier.

Understand how to use the '#' symbol as a comment.

4 Phase 2: initiating SnapMirror transfers

This phase uses only one of the helper scripts, 1_start_move.ksh.

Run it with the configuration file you built in the previous phase.

For example:

```
netapp@nidaros:~/netapp/scripts/Weekly_Aggr_Status> ./1_start_move.ksh tuesday_06_07_2010.cfg

Please run all of the commands in /dfm/netapp/scripts/Weekly_Aggr_Status/data/1_auto_clean.29320 after
validating that the output is correct.

Suggest using command : sh -x /dfm/netapp/scripts/Weekly_Aggr_Status/data/1_auto_clean.29320
```

Validate the output file and then run it. Use VI less or more to validate the generated command file, and then run it with the supplied command. The command you need to run as mentioned above is:

```
sh -x /dfm/netapp/scripts/Weekly_Aggr_Status/data/1_auto_clean.29320
```

This command calls the generated command script with the UNIX shell `sh` and shows each line as it executes. The script also adds entries into the helper script check_outstanding_jobs.

If you find errors in the command script, delete the generated command script, fix the errors in your configuration file, remove the lines that were recently added to the helper script check_outstanding_jobs, and then rerun 1_start_move.ksh.

The number after 1_auto_clean will be automatically generated by the DFM server. It is a unique number. Also, the section added to the check_outstanding_jobs script will have a notation to indicate when the section was added.

> Tip: Edit the configuration file you built, type # and a space at the beginning of each line. This helps you control which transfers you finalize later in Phase 4, allowing you to finalize one relationship at a time, even when you started multiple transfers.

The revised version of the file should look as follows:

```
#       Usage: 1_start_move.ksh Input_file
#
# File format (Space or comma seperated):
#  Cur_Backup_Filer Cur_vFiler Cur_Aggr New_Backup_Filer New_vFiler New_Aggr Volume Vol_Size
# eg-nasbkp-h01-mgmt ecomh1 aggr07 eg-nasbkp-f01-mgmt ecomf1 aggr02 sv_14_nv_wlnv12p_s01ora1_snap 5529g
# eg-nasbkp-h01-mgmt ecomh1 aggr07 eg-nasbkp-f01-mgmt ecomf1 aggr02 sv_14_at_cobaltinfa3p_s01ora1_snap 1200g
# eg-nasbkp-e02-mgmt ecome2 aggr23 eg-nasbkp-e02-mgmt ecome2 aggr35 sv_45_nv_cobaltmetadoc1c_s01ora1_snap 5655g
# eg-nasbkp-e02-mgmt ecome2 aggr23 eg-nasbkp-e02-mgmt ecome2 aggr38 sv_07_Infra_pv_mn2shrdea_vms_snap07 12737g
```

This essentially turns off the configuration line, since any line that starts with a # sign is a comment to be ignored by the scripts.

After completing this, go to the next phase.

4.1 Possible roadblocks

The following issues have been identified:

- Incorrectly identified vFiler units
- Destination volume size is smaller than source volume size

These are easy to fix. If necessary, break the configuration line off into its own configuration file and correct it before rerunning. You could also comment out all the lines that worked and just rerun it against the one that failed.

Remember to delete any duplicated lines from the check_outstanding_jobs script.

5 Phase 3: awaiting the completion of the initial SnapMirror baseline transfer

During this phase, the transfers in motion are monitored by running the helper script check_outstanding_jobs. Each of the transfers in motion will have a section detailing the current free space on the source and destination aggregates as well as a SnapMirror status.

Once a SnapMirror transfer becomes idle, immediately force an update. The basic form of the command is as follows:

```
rsh <PHYSICAL_FILER> vfiler run <VFILER> -S <SRC_VOL> <DEST_VOL>
```

The idea is to bring the lag time for the relationship down to a reasonable level. Note that this command is now automatically generated by 2_finalize_move.ksh for you.

5.1 Possible roadblocks

The following problems occurred during this phase:

- Destination aggregate filled up before transfer could complete. The new relationship has to be deleted.
- Destination aggregate filled up after the initial baseline transfer completed, but before the update could be forced.
- Source aggregate has too many snapshots, and was unable to take a new snapshot for the baseline transfer.

6 Phase 4: finalizing SnapMirror transfers

The final phase changes the SnapVault relationship and deletes the old volume. Five command scripts are generated - three of them can be run automatically and another two need interaction.

The output when running 2_finalize_move.ksh is shown below:

```
netapp@nidaros:~/netapp/scripts/Weekly_Aggr_Status> ./2_finalize_move.ksh tuesday_06_07_2010.cfg

Please run all of the commands in /dfm/netapp/scripts/Weekly_Aggr_Status/data/1_auto_clean.27369 after
validating the output is correct

        suggest using command sh -x /dfm/netapp/scripts/Weekly_Aggr_Status/data/1_auto_clean.27369

Please wait for snapvault updates to become idle.

Please run all of the commands in /dfm/netapp/scripts/Weekly_Aggr_Status/data/2_auto_clean.27369 after
validating that the output is correct.

          suggest using command: sh -x /dfm/netapp/scripts/Weekly_Aggr_Status/data/2_auto_clean.27369

Please MANUALLY run all of the commands in
/dfm/netapp/scripts/Weekly_Aggr_Status/data/3_auto_clean.27369 after validating that the output is
correct.

Please run all of the commands in /dfm/netapp/scripts/Weekly_Aggr_Status/data/4_auto_clean.27369 after
validating that the output is correct.

          suggest using command: sh -x /dfm/netapp/scripts/Weekly_Aggr_Status/data/4_auto_clean.27369

Please MANUALLY run all the commands in /dfm/netapp/scripts/Weekly_Aggr_Status/data/3_auto_clean.27369
after validating that the output is correct.
```

The first of the automatic programs is used to update the SnapMirror relationship before proceeding. After it is run, the status of the SnapMirror transfers can be checked with the check_outstanding_jobs script. Checking the status of the SnapMirror transfer is very important, because the update command returns before the SnapMirror transfer completes.

The output of the command returns the exact script names that need to be run. Any manual step will generate an error if the user attempts to run it anyway. Each script needs to be completed before the next task is started.

The automatic commands will run properly.

The manual steps involve running a command and then substituting parts of its output into the capital letter sections of the next part. There are slight differences between the necessary commands, depending on if the volume is being moved to a different aggregate on the current storage controller or to a different aggregate on a different storage controller.

The user should go through the manual steps one line at a time, and not keep track of the differences. All the differences are handled by the generating helper script.

See the following examples. These are taken from actual generated command files:

6.1 Setting the SnapVault Schedule

The exact commands will be automatically generated; however, an example is show below. This section will only be generated when a volume is moved to a different physical storage controller.

```
# rsh eg-nasbkp-h01-mgmt vfiler run ecomh1 snapvault snap sched sv_14_nv_wlnv12p_s01ora1_snap
# rsh eg-nasbkp-f01-mgmt vfiler run ecomf1 snapvault snap sched -x -o OPTIONS
sv_14_nv_wlnv12p_s01ora1_snap SNAPSHOT_NAME SCHEDULE
# rsh eg-nasbkp-f01-mgmt vfiler run ecomf1 snapvault snap sched sv_14_nv_wlnv12p_s01ora1_snap
# rsh eg-nasbkp-h01-mgmt vfiler run ecomh1 snapvault snap unsched sv_14_nv_wlnv12p_s01ora1_snap
```

Note: There is some word wrap in the above example.

The first line gives you the options, snapshot name to be used, and the snapshot schedule necessary for the second line. Cut and paste the necessary info and then run the second, third, and fourth lines.

6.2 Activating the New SnapVault Relationship

When a new SnapVault relationship is initialized with SnapMirror as we are doing here, there are a number of internal counters that need to be set. The only way to ensure that the internal counters are set correctly is to start the relationship.

This section of code will be automatically generated for all migrated relationships.

```
# rsh eg-nasbkp-f01-mgmt vfiler run ecomf1 snapvault status | egrep sv_14_nv_wlnv12p_s01ora1_snap
# rsh eg-nasbkp-f01-mgmt vfiler run ecomf1 snapvault start -S SOURCE NEW_DEST
# rsh eg-nasbkp-f01-mgmt vfiler run ecomf1 snapvault update -S SOURCE NEW_DEST
# rsh eg-nasbkp-f01-mgmt vfiler run ecomf1 snapvault status NEW_DEST
```

The first line gives you the source volume and the new destination volume. In the previous phases, there was some problem with DFM not having the correct vFiler name. The same problem exists in this step. If in the above example, the first line returned the destination qtree with `eg-nasbkp-f01:/vol/sv_14_winv12p_s01ora1_snap/1` then you need to correct the vFiler name to ecomf1, since it is the vfiler that we are running the command on. The new destination qtree name would be `ecomf1:/vol/sv_14_winv12p_s01ora1_snap/1`.

When you run the start command, the following error will be returned:

```
Snapvault configuration for the qtree has been set.
Qtree /vol/ sv_14_winv12p_s01ora1_snap/1 is already a replica.
```

This error is okay. Although an error is thrown, a number of internal counters are set properly that wouldn't otherwise be set. This is because the relationship was created by copying the volume that the qtree is part of, rather than rebaselining the relationship from scratch.

6.3 Deleting old Snapshot copies

This section provides information about deleting old Snapshots.

```
# rsh eg-nasbkp-f01-mgmt snap list -n sv_14_nv_wlnv12p_s01ora1_snap
# rsh eg-nasbkp-f01-mgmt "snap delete sv_14_nv_wlnv12p_s01ora1_snap SNAPSHOT"
```

The first line returns a list of Snapshot copies on the volume. Delete any Snapshot that does not follow the standard for the volume, or is busy.

In the following example, the highlighted Snapshot copies needs to be deleted. This example shows some of the other items to be cleaned up.

```
netapp@nidaros:~/netapp/scripts/Weekly_Aggr_Status> rsh eg-nasbkp-e02-mgmt snap list -n
sv_14_nv_kcsrchablenorm4a_s01ora1_snap
Volume sv_14_nv_kcsrchablenorm4a_s01ora1_snap
working...

date          name
------------  --------
May 27 10:45  ecome2(0118070165)_sv_14_nv_kcsrchablenorm4a_s01ora1_snap-base.1 (busy,snapvault)
May 27 10:00  ecome2(0118070165)_sv_14_nv_kcsrchablenorm4a_s01ora1_snap_new.2
```

```
May 26 20:15   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.0
May 26 20:12   ecome1(0118052219)_sv_14_nv_kcsrchablenorm4a_s01ora1_snap-base.1
May 25 20:16   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.1
May 25 15:51   ecome2(0118070165)_sv_14_nv_kcsrchablenorm4a_s01ora1_snap_new.1
May 24 20:14   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.2
May 24 20:10   ecome1(0118052219)_sv_14_nv_kcsrchablenorm4a_s01ora1_snap-base.0
May 23 20:12   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.3
May 22 20:13   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.4
May 21 20:12   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.5
May 20 20:17   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.6
May 19 20:20   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.7
May 18 20:19   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.8
May 17 20:14   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.9
May 16 20:11   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.10
May 15 20:13   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.11
May 14 20:19   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.12
May 13 20:14   sv_nv_kcsrchablenorm4a_s01ora1_snap_s01oradata1.13
Nov 13 00:01   nightly.0
Nov 12 00:01   nightly.1
Nov 11 00:01   nightly.2
Nov 10 00:01   nightly.3
Nov 09 00:01   nightly.4
Nov 08 00:00   nightly.5
Nov 07 00:01   nightly.6
Nov 06 00:00   nightly.7
```

Notice the nightly snapshots in the example. These are very old, and should be deleted. Deleting them will free up block space that is being reserved.

7 Configuration File Cleanup

This section provides information on how to cleanup your work to help keep track of what has been completed, and what still needs to be worked on.

Comment out the job you have completed in the configuration file. Use 5 # signs (#####) to indicate that you completed the job. An example follows:

```
#       Usage: 1_start_move.ksh Input_file
#
# File format (Space or comma seperated):
#  Cur_Backup_Filer Cur_vFiler Cur_Aggr New_Backup_Filer New_vFiler New_Aggr Volume Vol_Size
# eg-nasbkp-h01-mgmt ecomh1 aggr07 eg-nasbkp-f01-mgmt ecomf1 aggr02 sv_14_nv_wlnv12p_s01ora1_snap 5529g
##### eg-nasbkp-h01-mgmt ecomh1 aggr07 eg-nasbkp-f01-mgmt ecomf1 aggr02 sv_14_at_cobaltinfa3p_s01ora1_snap 1200g
##### eg-nasbkp-e02-mgmt ecome2 aggr23 eg-nasbkp-e02-mgmt ecome2 aggr35 sv_45_nv_cobaltmetadoc1c_s01ora1_snap 5655g
# eg-nasbkp-e02-mgmt ecome2 aggr23 eg-nasbkp-e02-mgmt ecome2 aggr38 sv_07_Infra_pv_mn2shrdea_vms_snap07 12737g
```

In the above example, two of the transfers are completed, and have been marked with extra # signs. The extra # signs have no effect on any of the helper scripts, the # signs have only been added for human readability.

8 Change control

Please note that this section is retained in case TRP's policies change and CRs become required in the future.

Most often, a change request (CR) number is created once every two weeks. During that time, move a number of volumes around to free up space.

Note: The exact entries, options and layout of the CR request page change from time to time without notice or warning. The information listed in this section is intended to be a guideline for filling out the form. Use your best judgment as you enter the data into the forms.

8.1 Change request information

There is a system to the CR creation. The following are intended as guidelines for building a CR. The actual fields for CRs change frequently. Ensure that all fields that have a red tick mark are filled.

The easy way to build a new CR is to copy an old CR. If for some reason you can't find an old CR, use the following default settings:

- Category: Storage
- Subcategory: Configuration
- Change Type: Modify
- Assignment Group: <Storage support - NetApp onsite>
- Assignee Name: <Your name>
- Preferred Start/End: <Pick a time in the future>
- Brief Description: Backup Filer Maintenance
- Employee ID: <Your c number goes here>
- Customer Impact During Change: None
- Validation/Checkout Plan: <Another team member>

8.1.1 Risk TAB

This section will have a number of drop down menus. The information entered here will be used to calculate the risk of the operation. Use the following choices, or the current closest match.

- Prod
- No outage: NO single point of failure during change
- None
- No Sys/Apps Users Impacted
- Familiar
- Backup Easy, Safe & Short (<30 min)

Click on the "Calculate Risk" hyperlink.

8.1.2 Asset TAB

- Asset ID: `eg-nasbkp-h01` (TAB out of the field and then everything else should be filled in automatically)

Note: Although multiple backup storage controllers will be worked on during the life of the CR, the CR form only supports one asset at this time..

8.1.3 Description

The following information should be inserted into the description field. The highlighted parts should be updated as appropriate:

```
This CR is intended to address changes necessary to maintain the thin
provisioning of the SnapVault environment. Requested changes include name
changes to correctly identify aggregates with available space and/or aggregates
which are loaded. Additionally, volumes may have been identified as needing to
be moved from extremely full aggregates to other less loaded aggregates.

The complete list of requested changes is as follows:
<Insert change list here>

This work will be completed by <Insert your name here> and will be checked out
by other members of the <Insert your group here>.
```

8.1.4 Business justification

The following should be inserted into the business justification field:

```
This work is intended to keep the SnapVault backup process working smoothly.
More space is allocated to the individual volumes than is actually available
inside the containing aggregates. This process allows the NetApp storage to be
highly utilized with little wasted disk space. Not performing these changes
could impact the ability to take SnapVault backups.
```

8.1.5 Back out plan

Put the following into the back out plan field of the CR:

```
Until the old volumes are actually deleted, the new locations simply contain
copies of the live data. If a need to back out is identified, the new volumes
should be brought offline and destroyed, with no impact to the existing
SnapVault backups.

If a need to back out a change is identified after the old volumes are
destroyed, the change would be backed out by following the same steps used to
move the destination volume to the new aggregate.
```

Although this isn't a detailed complete back out plan, it describes enough to satisfy the requirements of the CR.

9 Process checklist

The following is a high level overview of the entire process:

1. Open a CR for biweekly changes.
2. Run helper scripts to determine what volumes need to be moved.
3. Build a list of changes to be performed.
4. Build a configuration file.
5. Run 1_start_move.ksh with the configuration file.
6. Comment out all lines in the configuration file.
7. Check work progress with check_outstanding_jobs.
8. Update the SnapMirror relationship once initial transfer completes.
9. Uncomment jobs as they complete.
10. Run 2_finalize_move.ksh with the configuration file to generate next set of automatic and manual commands.
11. Run generated commands.
12. Comment out job in command file. Use 5 # signs (#####) to help keep track of completed jobs.
13. Comment out or delete commands to monitor a job in check_outstanding_jobs.

After the work is complete, close/request to close the CR.

10 Helpful Tips

Here are a list of things I do to help keep things flowing.:

14. Whenever I run the scripts “1_start_move.ksh” or “2_finalize_move.ksh” I redirect the output into a file named with the current day and the extension “.working”. For example, I might run the following: “`./2_finalize_move.ksh thursday_7_8_2010.cfg > thursday_7_8_2010.working`”. This allows me to keep track of what I’m currently working on.

11 Shift turnover

This section includes what prep work needs to be done for shift turnover as well when you receive turnover information.

11.1 Prep for turnover information

In order to successfully turnover the current state of the SnapVault secondary volume moves, a number of tasks must be completed:

1. Location of configuration files for all currently active volume transfers must be documented.
2. Script check_outstanding_jobs needs to be up-to-date with information about the currently running jobs.
3. Notate the currently active jobs.
4. Notate any failed jobs or transfers that had to be canceled.

11.2 Receiving turnover information

When turnover information is received, there are a number of tasks that need to be done before starting additional volume moves:

1. Review output of check_outstanding_jobs to be aware of all currently running move jobs.
2. Review configuration files to validate that all entries are commented out.
3. Review configuration files to validate that all active jobs have a commented out configuration entry.
4. Review the output of fullness.ksh to be aware of any critical space issues.
5. Review the open jobs list.

12 Final notes

The goal of this document is to cut out 90% of the work that would be prone to error. The last 10% would be very difficult to code and prone to having all sorts of exceptions. This final 10% is easy for anyone to look at, comprehend and then run the necessary work.