Versant High Availability Backup Usage Manual

Release 7.0.1.4

VERSANT

Versant History, Innovating for Excellence

In 1988, Versant's visionaries began building solutions based on a highly scalable and distributed object-oriented architecture and a patented caching algorithm that proved to be prescient.

Versant's initial flagship product, the Versant Object Database Management System (ODBMS), was viewed by the industry as the one truly enterprise-scalable object database.

Leading telecommunications, financial services, defense and transportation companies have all depended on Versant to solve some of the most complex data management applications in the world. Applications such as fraud detection, risk analysis, simulation, yield management and real-time data collection and analysis have benefited from Versant's unique object-oriented architecture.

For more Information please visit **www.versant.com**

Versant US
Versant Corporation
255 Shoreline Drive, Suite 450, Redwood City, CA 94065
Ph +1 650-232-2400, Fx +1 650-232-2401

Versant Europe
Versant GmbH
Wiesenkamp 22b, 22359 Hamburg, Germany
Ph +49.40.60990-0, Fx +49.40.60990-113

The information contained in this document is a summary only.

For more information about Versant Corporation and its products and services, please contact Versant Worldwide or European Headquarters.

Table of Contents

CHAPTER 1

Introduction

This Chapter gives detailed introduction of “Habackup”.

The Chapter covers the following in detail:

- Overview
- Habackup Features

OVERVIEW

Increasingly, big enterprises are deploying Versant Object Database in mission critical applications. Typically, such customers use mirrored disk devices for high performance and fault tolerance. Versant database administrators are seeking for innovative ways for activities like,

- fast database backup/restore without affecting the server availability
- duplicate production databases as a standby for decision support systems (DSS)

Versant **H**igh **A**vailability Backup (**habackup**) solution is a generic way of achieving continuous online backup of Versant Database. It exploits the capabilities of special storage devices. It allows the user to execute certain operations such as, splitting a mirrored device, after bringing the database to a consistent state. Thus, a copy of the database can be made at a consistent point without jeopardizing server availability, data integrity and performance. Subsequently, this copy can be used for various purposes including, continuous online backup and stand-by database for DSS applications.

The `habackup` solution is an extension of `vbackup` mechanism that exploits the device mirroring functionality of the modern disk arrays.

For more information, on the `vbackup` utility, please refer to Chapter "Database Utilities" of *Versant Database Administration Manual*.

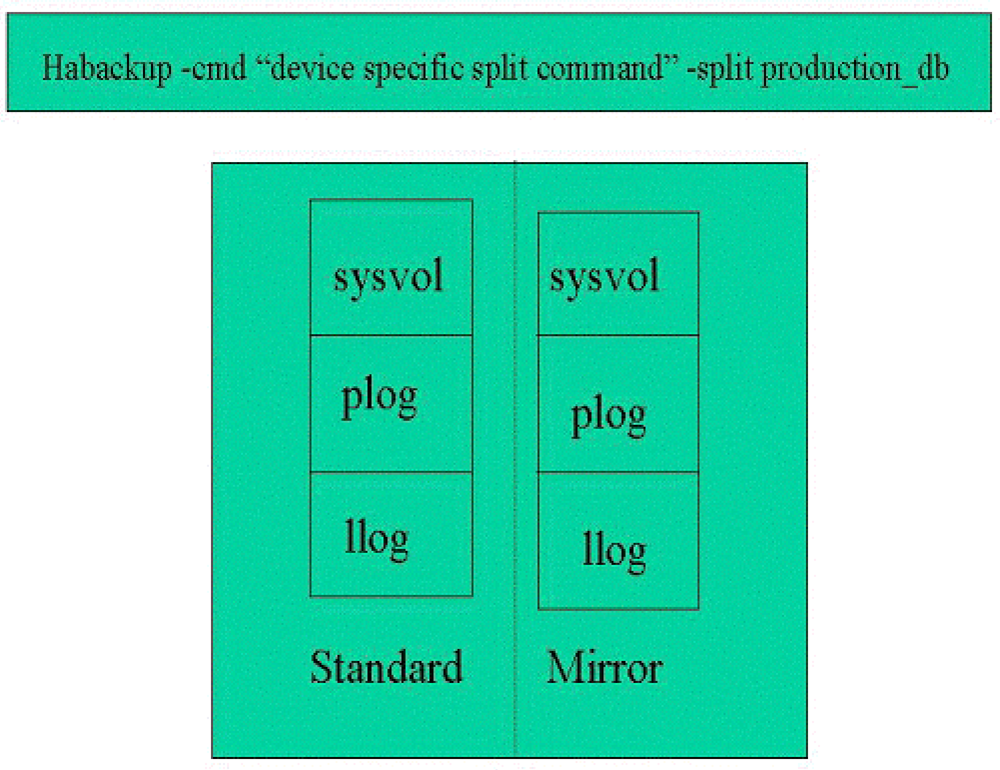
HABACKUP FEATURES

The Versant `habackup` solution offers the following:

Ability to Perform Split

A device pair consists of a mirror device and a standard device. The standard device (on which production database resides) can have any mirror structure (unprotected, RAID, RAID with SRDF). After configuration and initialization of a mirrored disk array, the mirror devices contain no data. The device pair should be synchronized so that both the standard and mirror devices contain the same data. Typically the device vendors provide the ability to synchronize and split mirrored pair.

Figure 1 illustrates a Versant object database [say, production_db] in operation. The system, physical log and logical log volumes are shown to be residing on mirrored disk array.



```
Habackup -cmd "device specific split command" -split production_db
```

Figure 1

A split operation is carried out by using the `-split` command of `habackup`, which ensures that the appropriate timestamp fields on the database are updated and at the same time it is in a consistent state before the split operation is issued. The (device) vendor-specific command to be executed for a split operation is indicated by the `-cmd` option to `habackup`. `Habackup` executes this command or even a script and waits for it to finish. It is assumed that the vendor specific split command or script would return an error if it is unsuccessful. Based on the error returned, `habackup` will either complete or abort the split. The database image on the mirror device, obtained after the split operation, is a consistent snapshot of the database, which can be archived to a backup media. The "`-split`" command of `habackup` is synonymous to level 0 backup of `vbackup`.

Level 1 and Level 2 Backups

Habackup also offers the ability to perform incremental backups. After performing split it is possible to take level 1 or level 2 backups of the production database.

Roll-forward Archiving

Roll-forward archiving can be enabled with `-roll` option during split or level 1/2 backups. Similar to `vbackup` utility, `-log` option can be used to start the archive process and `-off` to stop it.

Restore

In the event of a crash, the standard device can be restored by forming device pairs (operation against of split) using the commands provided by device vendors. After the device level restore takes place, the standard device contains the snapshot of the database at the time of split. The level 1/2 backups and RF archives of the production database after the split, are useful at this point to bring the database to the state just prior to crash.

In the subsequent part of this document, we present Versant Habackup and EMC Timefinder integration with syntactic details of the habackup commands and options.

Warm Standby (Incremental Restore)

A Warm Standby allows to incrementally restore the Roll-forward archives after the habackup has restored the standard device. I.e. every roll forward archive, that has been created by archiving the database transactions of the original database, can be applied to the Warm Standby database any time, but exclusively in the correct sequence.

This way there is always an "almost up-to-date" state of the database available, a so called Warm Standby. The Warm Standby is being maintained by applying the most recent roll forward archives.

In the event of a failure only the last written archive and the logical.log file need to be applied to the Warm Standby database in order to make it up-to-date.

CHAPTER 2

Use of Habackup with EMC Timefinder

This Chapter gives detailed explanation on the “Use of Habackup with the EMC Timefinder”.

The Chapter covers the following in detail:

- Symmetrix Disk Devices and EMC Timefinder
- Setting up an EMC Symmetrix system
- Using Habackup
- Use Cases
- Restrictions
- Summary of Habackup Commands and Options

SYMMETRIX DISK DEVICES AND EMC TIMEFINDER

The Versant habackup solution is a way of achieving scalable online backups of a Versant database, by exploiting the online mirroring capabilities of an EMC Symmetrix system. The new utility `habackup` enables "hot" database backups in conjunction with the mirroring capabilities of an EMC Symmetrix system.

EMC products and terminology are described on the EMC web site at http://www.emc.com. EMC TimeFinder is a unique EMC software residing on both the host/server and Symmetrix systems, which allows system and storage administrators to create, in background mode, independently addressable **B**usiness **C**ontinuance **V**olumes (**BCV**s) for mainframe and open system information storage. The BCVs are mirror images of active production volumes, which can be used to run simultaneous tasks in parallel with one another. This parallel processing capability, known as workload compression, allows for significantly increasing company efficiency and productivity while maintaining continuous support for the needs of the enterprise. Once BCVs have been created, they can be split from their production volumes and used for such tasks as loading data warehouses, testing new applications, running batch jobs, backups and other functions that typically mean a disruption to normal business support. With EMC TimeFinder, business keeps running at full capacity even as these other scheduled outages occur. Once the task on the BCV is complete, the volume can be re synchronized with the production volume, reassigned to another production volume, or maintained as is for another task.

Habackup interacts with EMC Timefinder using **S**ymmetrix **C**ommand **L**ine **I**nterface (**SYMCLI**) and it takes advantage of the storage capabilities of EMC. The Versant `habackup` utility command “`-split`” can be used to execute the SYMCLI split command. The SYMCLI split is provided as an argument to the habackup command option. It splits the BCV pair from the standard Symmetrix device without affecting online operations on the standard device for regular database operations from their hosts. It is assumed that the database volumes reside on the same Symmetrix Disk Array and control files reside on a file system, which is not stored on the disk array.

EMC TimeFinder introduces the idea of device groups. These groups may be treated like devices. The TimeFinder synchronization and split commands will be applied to these device groups as if it were devices. For `habackup` it is necessary to build one “database device group”. The devices in this group must be raw devices for a hot (on-line) split operation [Refer to the section "Restrictions" of this chapter]. All the three volumes of a database, namely, system, logical log and physical log, should be created on this device group. If the device group has more than three devices, it is possible to store multiple databases within it.

The original standard device group and its BCV copy (the BCV device group) build the BCV pair. The `habackup` utility operates by first bringing the database to a consistent state and then allowing the mirrored database image to be taken off-line by calling an external program or script. A typical backup `-split` command would be:

```
habackup -cmd "symmir -g versantDevGroup split" -split production_db
```

Where "`symmir -g versantDevGroup split`" is the SYMCLI command which puts an EMC Symmetrix mirrored disk (also known as a business continuance volume - BCV) off-line. "`production_db`" is the name of the database.

Versant Habackup is certified to run with the CLARiiON CX product family and EMC PowerPath.

SETTING UP AN EMC SYMMETRIX SYSTEM

A host system connected to an EMC-symmetrix unit is typically configured with a set of logical and physical devices. These devices can be either Standard or BCV devices. Standard devices are mirrored onto (backup) BCV devices. But, before a mirroring pair can be established, the devices, which participate in the mirroring, must be grouped together into what is called a "device group". The (sample) steps followed to archive such a group are described below.

Finding names of the available devices

First we need to find out the names of the logical and physical devices, which are visible to the host. Notice that SYMCLI commands can be executed only by the operating system user `root`.

Symmetrix host database, can be rebuilt by:

```
symcfg discover
```

A listing of the logical devices can be obtained by:

```
symdev list
```

A listing of the physical devices can be obtained by:

```
sympd list
```

Creating a device group

Next, create a device group.

```
symdg create versantDevGroup
```

(where `versantDevGroup` is the device group name)

Confirm that the group has been created.

```
symdg list
```

(lists all the groups)

Or

```
symdg show versantDevGroup
```

(list all the details of a group)

Adding standard and mirror devices to a group

Then, the Standard and the BCV devices are grouped together as parts of the device group, to form a mirroring pair.

Suppose the physical and logical names of disk devices are of the form `/dev/rdsk/c*t*d*s*` and `DEV*`, respectively.

Add standard devices to the device group.

```
symld -g versantDevGroup add pd /dev/rdsk/c2t0d1s2
symld -g versantDevGroup add pd /dev/rdsk/c2t0d2s2
symld -g versantDevGroup add pd /dev/rdsk/c2t0d3s2
```

Associate BCV devices (configured as physical devices here) with the standard devices in the group.

```
symbcv -g versantDevGroup associate pd /dev/rdsk/c2t0d11s2
symbcv -g versantDevGroup associate pd /dev/rdsk/c2t0d12s2
symbcv -g versantDevGroup associate pd /dev/rdsk/c2t0d13s2
```

It is possible to add or associate devices using logical names instead of physical names.

```
symld -g versantDevGroup add dev DEV001
```

Or

```
symbcv -g versantDevGroup associate dev DEV011
```

Note that, it is not necessary to associate the BCV devices with the standard devices before creation of the database. It is possible to associate them after database has been created. However, it is essential to establish the device pairs after BCV devices are associated, so that the mirror images will get a copy of data on the corresponding standard devices. All device pairs in a group can be established using:

```
symmir -g versantDevGroup establish
```

It is also possible to establish a pair(s) incrementally by specifying logical device names.

```
symmir -g versantDevGroup establish DEV001 DEV002
```

Incremental establish/restore is very useful if multiple databases reside on a single device group and only one database needs to be backed-up or restored.

The list of standard and BCV devices in the group may be verified using:

```
symdg show versantDevGroup
```

USING HABACKUP

After forming the group, makedb and createdb commands can be used to create a database on the newly formed Symmetrix device group. Versant configuration file profile.be is created in VERSANT_DB directory after makedb. The following changes are made to the profile.be for mapping the versant database volumes to the Symmetrix devices.
Assuming that the devices are big enough, each of them is configured to store versant volume of size of 3072M as shown below.

profile.be settings

Creation Parameters

These are used by CREATEDB to create the database volumes and used by STARTDB to find the volumes.

sysvol

This is the size and location of the system volume. The volume will be expanded if necessary if it is a file, but not if it is a raw device.

For e.g; entry in profile.be file for this parameter will look like:

```
sysvol           3072M          /dev/rdsk/c2t0d1s2
```

where 3072 is the size of the system volume and /dev/rdsk/c2t0d1s2 specifies the location.

You cannot change the size or name once a database has been created. You can add additional space with the addvol utility.

A guideline for setting the size of sysvol is:

guideline = min_size+(1.25*(avg_o_size+40)*num_objects)

where avg_o_size is the average size of objects in the database and num_objects is the number of objects.

For the default extent_size (2), the minimum volume size is 1800K and the maximum size is 2047M.
The absolute maximum is platform dependent.

plogvol

This is the size and location of the physical log volume. The volume will be expanded if necessary if it is a file, but not if it is a raw device. The minimum size is 2M.

For e.g; entry in profile.be file for this parameter will look like:

```
plogvol                  3072M           /dev/rdsk/c2t0d2s2
```

where 3072 is the size of the system volume and `/dev/rdsk/c2t0d2s2` specifies the location.

You cannot change the size or name once a database has been created.

A guideline for setting the size is:

guideline = 32K+(2*max_o+100)*num_o_per_tr*num_tr

where `max_o` is the size of the largest object in the database, `num_o_per_tr` is the largest number of objects that will be involved in a transaction and `num_tr` is the maximum number of concurrent transactions.

llogvol

The size and location of the logical log volume. The volume will be expanded if necessary if it is a file, but not if it is a raw device. The minimum size is 2M.

For e.g; entry in profile.be file for this parameter will look like:

```
llogvol                  3072M           /dev/rdsk/c2t0d3s2
```

where 3072 is the size of the system volume and /dev/rdsk/c2t0d3s2 specifies the location.

You cannot change the size or name once a database has been created.

A guideline for setting the size is:

guideline = 32K+(2*max_o+100)*num_o_per_tr*num_tr

where max_o is the size of the largest object in the database, num_o_per_tr is the largest number of objects that will be involved in a transaction and num_tr is the maximum number of concurrent transactions.

During normal operations of the database, the Standard devices and the BCV devices are mirrored or paired together. [However, it is possible to maintain only selected volumes on the Symmetrix disk array and the remaining volumes on any other filesystem.] Any data written to the Standard device will automatically be replicated on the BCV device, if this paring has been established.

Backup / Split

A (full) backup of the database is achieved by "splitting" the Standard-BCV device pair. A typical `split` command looks like this:

```
habackup -cmd "symmir -g versantDevGroup split" -split production_db
```

Where "`symmir -g versantDevGroup split`" is EMC command, which stops the mirroring process and takes the BCV off-line. Since this command internally invokes SYMCLI command, it must be executed by OS user `root`.

The split command with RF enabling has the syntax:

```
habackup -cmd "symmir -g versantDevGroup split" -roll -split
 production_db
```

Level 1 Backup

A level 1 backup (after a "split" had already taken place) can be performed using:

```
habackup -level 1 -device /dev/level1tape [-roll] -backup
 production_db
```

Level 2 Backup

Similarly a level 2 backup (after a level 1 backup) can be performed using:

```
habackup -level 2 -device /dev/level2tape [-roll] -backup
 production_db
```

Backup/Split in FTS Environment

A full backup of FTS database can be achieved by using habackup `–split` command in conjunction with ”`-startsync`” option.

Prerequisites:

The synchronization records should be removed from the database and the database should be in the `SUSPEND` state. This can be achieved by running “`ftstool –stopsync`”.

Once the backup has been completed, the specified FTS database will start accumulating the synchronization records and will be put in AWAITING_RESTORE state. See the description of the ftstool utility for an explanation of the fts database states and a typical usage scenario.

For example:

Suppose FTS database names are “primary_db” and “secondary_db”. So for restoring the failed replica database (say secondary_db), the DBA needs to take a backup of an FTS database “primary_db” and the command will be as follows:

```
habackup –cmd “symmir –g versantDevGroup split” –split –startsync
 primary_db
```

Roll Forward Archiving

Enable Roll forwarding

It is possible to use -roll option to enable roll forwarding during split or level 1 or 2 backups.

For example:

Roll forwarding can be enabled during level 1 backup as:

```
habackup -level 1 -device /dev/level1tape -roll -backup production_db

VERSANT Utility HABACKUP Version 7.0.1.3
Copyright (c) 1989-2006 VERSANT Corporation
Backing up database `production_db' to device `/dev/level1tape':
0%             50%            100%
|          |         |         |        |
........................................
Backup has completed successfully.
```

Start Roll forward archiving process

```
habackup -device /dev/rolltape -log production_db

VERSANT Utility HABACKUP Version 7.0.1.3
Copyright (c) 1989-2006 VERSANT Corporation
Press <return> when you are ready to exit.
Archiving log records to device `/dev/rolltape':
Database `production_db' is now being archived.
```

`/dev/rolltape` is the tape device where logical log records are archived during RF arching.

Stop Roll forward archiving process

The command to turn roll forward archiving off for the specified database is as follows:

```
Habackup –off dbname1 [dbname2….]
```

By turning roll forward archiving off, the roll forward archive chain gets interrupted, i.e. at restore time you can restore the database only up to the point when you turned roll forward archiving off.

This indicates that the log records will not be written to the roll forward archive and may be discarded when the log file space is reused.

Only together with that backup the archive files can restore the database up to the last transaction (or up to the point when rollforward has been switched off).

Restore

Restoring the database involves transferring the database image on the BCV back to the Standard device. This can be achieved as:

```
symmir -g versantDevGroup restore
```

After the BCV device has been restored back onto the Standard device, the device group is in synchronized state. At this point the database restore needs to be performed so that available level 1/ 2 backups and RF archives can be applied. The syntax for restoring databases is:

```
habackup -restore production_db
```

This command prompts the user for next level backups and RF archives, if any. However, if the user doesn't supply level 1/2 backups at this point, it is possible to apply them later using command:

```
habackup -level 1 -device /dev/level2tape -restore production_db
```

Or

```
habackup -level 2 -device /dev/level2tape -restore production_db
```

Note that it is essential to apply level 1 backup before level 2 backup can be applied.

During an habackup restore operation, it is possible to apply the records logical.log file that has not been archived. In order to do this, a physical copy of logical.log file must be made and kept on a safe location before doing an EMC device level restore. This is necessary to ensure that the current logical.log file is not overwritten when the devices in a device group are restored. The database must be stopped before the current logical.log file is copied.

If roll forward archiving is turned on at the time of the last backup; the habackup restore procedure will prompt if the user wants to apply the current logical.log records to the restored database. If the user chooses to apply the copy of the logical.log the user must specify the location of the copied logical.log file.

Full restore on all the devices in a device group, i.e. on all the databases stored on the group, is performed using:

```
symmir -g versantDevGroup -full restore
```

Incremental restore can be useful in the situations where multiple databases are stored on the same device group. As an example, suppose after performing the split, one or more but not all the databases on the standard devices crash. Then it is possible to restore incrementally the devices pertaining to the crashed databases using:

```
symmir -g versantDevGroup -full restore DEV001 DEV002 DEV003
```

Then a level 1 backup and the RF archive is restored on the database:

```
habackup -restore production_db


VERSANT Utility HABACKUP Version 7.0.1.3
Copyright (c) 1989-2006 VERSANT Corporation
Restore has completed successfully.
Would you like to do another level of restore on database
 `production_db'? [ default = no ]y
```

```
Current settings are:
device   = `dumphost:/dev/nrmt8'
position = `0'
capacity = `dynamic'
blocking = `10 Kilobytes'
Mount the appropriate tape and press return to begin... [?=help]
dev /dev/level1tape
Current settings are:
device   = `/dev/level1tape'
position = `0'
capacity = `dynamic'
blocking = `10 Kilobytes '

Change additional settings or type <return> to proceed. [?=help]
Restoring database `production_db' from device `/dev/level1tape':
0%                  50%                100%
|         |          |         |        |
........................................
Restore has completed successfully.

Would you like to do another level of restore on database
 `production_db'? [ default = no ]n
Current settings are:
device   = `level1tape'
position = `0'
capacity = `dynamic'
blocking = `10 Kilobytes '

Insert log archive #1 of database `production_db'. [?=help]
dev /dev/rolltape
Current settings are:
device   = `/dev/rolltape'
position = `0'
capacity = `dynamic'
blocking = `10 Kilobytes '

Change additional settings or type <return> to proceed. [?=help]
Current settings are:
device   = `/dev/rolltape'
position = `0'
capacity = `dynamic'
```

```
blocking = `10 Kilobytes '

Insert log archive #2 of database `production_db'. [?=help] quit

Do you have a copy of logical.log file that you would like to apply at
 this point ? [default = yes] yes

Enter the path of logical.log : /tmp/logical.log
```

After the level 0 restore (i.e. the restore from splitted BCV) completes, `habackup` will prompt for additional levels of restore. In this example, we input "no" after level 1 (we've only taken a level 1 backup!) which will cause the program to prompt for a log archive. The log archive tape is inserted at this point and the device option updated to point to the tape containing the archive (`/dev/rolltape` in the example).

After you are done with inserting the last log archive (`/dev/rolltape` in this case), it is possible that there were home transactions in logical log that could not be archived to `/dev/rolltape`. In this case, you need to apply the records from saved logical log to ensure that the database is restored to the last committed transaction. If you had made a copy of the existing logical.log (before doing the restore operation), the path must be provided at this point.

Once the restore procedure has completed and the database is up and running, it is safe to remove the temporary copy of the logical log file `/tmp/logical.log`.

Restore in FTS Environment

Vedding online restore is possible by using the Habackup `–restore` command in conjunction with the `-synchwith` option.

Prerequisites:

- The crashed replica database should be removed.
- The replica database directory and the profile.be file should be recreated with `makedb` utility.
- The BCV split device of the up database should be copied to the machine that hosts the crashed replica database. The `profile.be` of the replica database should be edited to map the database volumes copied from the BCV device.
- Make sure you do not use the `.lock` file of the original/up database as it may contain stale FTS related information which can create problems during habackup restore. It is advised to recreate the `.lock` file using "`dbinfo -c`" command before invoking "habackup -restore" command for FTS database.

For example;

In order to restore “secondary_db”, the command will be as follows:

```
habackup –restore secondary_db –syncwith primary_db
```

In this case, Habackup will first check if the restore database name (i.e. secondary_db) matches with the replica database name that is stored in the physical log header of the BCV split device. Only if the two database names match will Habackup proceed, otherwise, an error `E12439, HABACKUP_OLIB__INCONSISTENT_RESTORE`, will be returned.

On restoration, re-synchronization will be initiated by starting the polling process on the `–syncwith` database (i.e. primary_db) and once it’s complete, the state of both the replica-databases will be set to “Syncing Done”.

Warm Standby (Incremental Restore)

Warm Standby i.e., Incremental restore strengthens Versant's high availability capabilities.

A combination of habackup and Incremental Restore is the fastest possible way to recover a Versant database without loosing a single committed transaction.

The Warm Standby feature of Versant's habackup is in it's core the same like the one for vbackup. It allows to apply rollforward archives, that have been taken from the original database since the last habackup, to an habackup restored database incrementally.

Features:

The purpose of Warm Standby is:

- Whenever the restore process requests a new rollforward archive file, the restore process can be interrupted by a suspend (rather than a quit).
- Whenever the restore process gets suspended, the restore process can be resumed again by applying the next available rollforward archive files and/or the logical.log file.
- Between suspend and resume process, the database is in "Restore suspended (unstartable) mode", i.e. only the habackup -resume is allowed to start and access this database.
- In case of an emergency the restore process needs only to be resumed with the last roll forward archive file(s) and/or the logical.log of the primary database in order to have the last transactions recovered in the Warm Standby database that is now ready to be used as the new primary database.

This utility requires a separate license.

Examples:

For backup: `habackup -roll -cmd "split.sh <dbname>" -split <dbname>`

For restore: `habackup -restore <dbname>`

For resume: `habackup -dev <roll forward archive name> -restore <dbname> -resume`

For more information, please refer to the *Warm Standby Usage Guide.*

USE CASES

The flexibility provided by SYMCLI commands and Versant habackup solution makes a variety of use cases possible. We describe four different scenarios below.

Case 1. Habackup as a backup solution

The purpose of any backup solution is recovery after crash to the point as close as possible to the last committed transaction. Versant `vbackup` solution is useful for incremental level 0, 1 and 2 backups (on tapes or filesystems) in conjunction with roll-forward archiving. Similarly habackup solution can be used for recovery up to the point of last archived record of the logical log.

As an example consider the device group versant and a database production_db residing on this group. A split operation on this group is synonymous to a level 0 backup where the backup media is the set of mirror devices. Assume the following sequence of events:

a. `habackup -cmd "symmir -g versantDevGroup split" -split production_db`

some `update/delete/inserts` on the database

b. `habackup -cmd "symmir -g versantDevGroup split" –split -startsyn production_db`

some `update/delete/inserts` on the FTS database...

c. `habackup -dev /temp_device/dev1 -level 1 -backup production_db`

some `update/delete/inserts` on the database...

d. `habackup -dev /temp_device/dev2 -level 2 -backup production_db`

e. `habackup -dev /log_device/log1 -log production_db` (in a different command window).

some update/delete/inserts on the database...

f. crash! The database on the standard device crashed!

Recovery may be performed by following steps

g. `symmir -g versantDevGroup restore`

h. `symmir -g versantDevGroup syncwith`

i. Make sure that the devices are in synchronized state using SYMCLI command -

```
symmir -g versantDevGroup verify
```

j. Once all the devices are in synchronized/restored status after restore command, perform database restore using-

```
habackup -restore production_db
```

This command has a user interface which requests for levels 1 and 2 backup devices and the roll-forward archive logs.

Case 2. Habackup as a group backup and selective restore solution

Assume that the device group versant contains six standard devices and six BCV devices. It is possible to use two sets of three devices each for storing two databases. If both the databases crash after a split, the steps in **Case 1** can be used to restore both of them with an exception of step h. The step h should contain names of both the databases to be restored.
Note also that the "`-split`" option of `habackup` should be followed by names of both databases.

However, if only one database (`production_db1`) crashes, it is possible to restore it independent of the other. In this situation, the incremental restore feature of symmir command should be used followed by "`habackup -restore production_db1`".

Case 3. Standby database using Habackup for Analytical Processing

It is possible to start the database from the mirror image with a different database name and use it either for performing usual level 0 backup, using `vbackup` utility or analytical processing. If level 1/2 backups and RF archives of the production database (which is on the standard devices) are available when the mirror image gets restored, it is possible to apply these backups and play the RF archives on the mirror image.

As an example consider that steps a to d from **Case 1** are performed for database production_db. Hence the mirror of this database snapshot (at the time of split) is available along with the backups and RF archive. The database on the mirror image can be started as follows:

a. `makedb production_db_mirror`

b. edit `profile.be` to map the database volumes to the BCV devices.

c. `habackup -restore production_db_mirror`

This makes a new `osc-dbid` entry. Also it asks for incremental backups and RF archives similar to usual restore process.

Notice that, no `createdb` has been executed. It would be fatal to run `createdb` after editing the `profile.be` to make use of existing mirror database.

Case 4. Restoring a replica database

As an example consider the device group versant and a replica databases ‘`primdb`’ and ‘`repdb`’ residing on this group. Suppose the failed database has been destroyed or polling is taking too long to re synchronize the two databases, in such cases you might want to do the following:

1. Remove the failed database and use `makedb` utility to create a new database to be used for replication.

   If you create a new database, its name should be the same as the failed database.
   For example, suppose the two replica database names are `primdb` and `repdb` and in this scenario `primdb` database is UP and running while `repdb` has failed.

- Remove the failed database (i.e. repdb) and its directories:

  ```
  removereplica -rmdir repdb
  ```

- Recreate the directories and database files, using the same name as the database that has failed:

  ```
  makedb repdb
  ```

2. Stop accumulation of synchronization records on the running database by using `ftstool` utility:

   ```
   ftstool -stopsync primdb
   ```

   This will change the status of the running database i.e. `primdb` from "`POLLING`" or "`SYNCING`" to "`SUSPEND`" and discard accumulated resynchronization records.

   If the specified database is in any state other than "`POLLING`" or “`SYNCING`” when this option is used, the error `UT_DB_WRONG_STATE` will be returned.

3. Back up the running database using habackup `–split –startsync` option

   For example:

   ```
   habackup -cmd "symmir -g versantDevGroup split" –split -startsync
   primdb
   ```

   Using the above options, the database changes its state from `SUSPEND` to `AWAITING_RESTORE`. However, if the status of the running database is not `SUSPEND` when the habackup command is executed, then the error `UT_DB_WRONG_STATE` is thrown.

   If an error occurs during backup process, step 1 (`ftstool -stopsync`) has to be replayed.

4. Restore the database using the back up taken on the BCV device, generated in step 3 by using the habackup `–restore -syncwith` option

   For example,

   ```
   habackup -restore repdb -syncwith primdb
   ```

   The above command will automatically start polling and re synchronize the ‘`repdb`’ database, when the restore process has been completed.

RESTRICTIONS

- Our customers using Veritas filesystem with EMC devices would experience the following limitation. In order to perform split operation the file system needs to be unmounted. Suppose that a filesystem is created on the disk array and it is used to store logical log. Then the filesystem cannot be unmounted in order to perform a split because the logical log is constantly updated when the database is active. Hence a split cannot be performed on an active database. The only alternative in this situation is to perform split on an off-line database.
- EMC SYMCLI commands are allowed to be executed only as an OS user `root`. Traditionally, Versant allows the administrative commands to be run only as a `DBA`. Versant's split command used for EMC devices is:

  ```
  habackup -cmd "symmir -g versantDevGroup split" - split <dbname>
  ```

  It internally invokes the SYMCLI command for split. In order to ensure correct execution of EMC's split command, the habackup with `-split` option is allowed to be run as a root. It should be noted that no other option of `habackup` will be allowed to be run by the OS user `root`.

SUMMARY OF HABACKUP COMMANDS AND OPTIONS

The Versant High Availability Backup utility (habackup) supports the following command and option parameters:

```
habackup [options] command
```

The alternatives for the `option` parameters are:

`-cmd`

Executable command to be executed to achieve a "`split`" or its equivalent operation.

`-level [1 | 2]`

The incremental level of backup to perform. Use of Level 0 is not allowed with "`–backup`" command.

`-device devicename`

The backup device on which to read or write. Device specification is not necessary for level 0 "`-restore`".

`-rollforward`

When used in conjunction with the `–backup` or `–split` command, turn roll forward archiving on but do not begin writing to the rollforward log.

The alternatives for the `command` parameters are:

`-split [-startsync]`

Backup one or more local databases as level 0 backup using the specified command.
The `–startsync` option should only be used when the FTS database is backed up.

`-restore dbname1 [dbname2...]  [ -syncwith <dbname> ]`

Restore one or more databases specified. When restoring from a mirror device, there is no need to specify the source device or level. By specifying device or level this option can be used for applying incremental backups during recovery.

The `-syncwith` option is to be used only when FTS database is being restored.

-dev roll forward_device -restore dbname -resume

The resume as part of the Incremental Restore feature needs a device name specified. It has to be the roll forward archive next to the one that has been applied last after which the restore has been suspended. If only a logical.log file needs to be applied to the warm standby database then the keyword "llog_only" has to be used as a device name:

-dev llog_only -restore dbname -resume

For more details please refer to the Warm Standby Usage Guide. The description there applies to the habackup accordingly.

`-log dbname1 [dbname 2 …]`

Start writing log records for the specified databases to the rollforward archive file.

`-off dbname 1 [dbname 2 ...]`

Stop writing log records for the specified databases to the rollforward archive file.

Index