

Thin Image Advanced Snapshots for Active PostgreSQL Database

Best Practices Guide

© 2025 Hitachi Vantara LLC. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including copying and recording, or stored in a database or retrieval system for commercial purposes without the express written permission of Hitachi, Ltd., Hitachi Vantara, Ltd., or Hitachi Vantara LLC (collectively "Hitachi"). Licensee may make copies of the Materials provided that any such copy is: (i) created as an essential step in utilization of the Software as licensed and is used in no other manner; or (ii) used for archival purposes. Licensee may not make any other copies of the Materials. "Materials" mean text, data, photographs, graphics, audio, video and documents.

Hitachi reserves the right to make changes to this Material at any time without notice and assumes no responsibility for its use. The Materials contain the most current information available at the time of publication.

Some of the features described in the Materials might not be currently available. Refer to the most recent product announcement for information about feature and product availability, or contact Hitachi Vantara LLC at https://support.hitachivantara.com/en_us/contact-us.html.

Notice: Hitachi products and services can be ordered only under the terms and conditions of the applicable Hitachi agreements. The use of Hitachi products is governed by the terms of your agreements with Hitachi Vantara LLC.

By using this software, you agree that you are responsible for:

1. Acquiring the relevant consents as may be required under local privacy laws or otherwise from authorized employees and other individuals; and
2. Verifying that your data continues to be held, retrieved, deleted, or otherwise processed in accordance with relevant laws.

Notice on Export Controls. The technical data and technology inherent in this Document may be subject to U.S. export control laws, including the U.S. Export Administration Act and its associated regulations, and may be subject to export or import regulations in other countries. Reader agrees to comply strictly with all such regulations and acknowledges that Reader has the responsibility to obtain licenses to export, re-export, or import the Document and any Compliant Products.

Hitachi and Lumada are trademarks or registered trademarks of Hitachi, Ltd., in the United States and other countries.

AIX, DB2, DS6000, DS8000, Enterprise Storage Server, eServer, FICON, FlashCopy, GDPS, HyperSwap, IBM, IntelliMagic, IntelliMagic Vision, OS/390, PowerHA, PowerPC, S/390, System z9, System z10, Tivoli, z/OS, z9, z10, z13, z14, z15, z16, z17, z/VM, and z/VSE are registered trademarks or trademarks of International Business Machines Corporation.

Active Directory, ActiveX, Bing, Excel, Hyper-V, Internet Explorer, the Internet Explorer logo, Microsoft, Microsoft Edge, the Microsoft corporate logo, the Microsoft Edge logo, MS-DOS, Outlook, PowerPoint, SharePoint, Silverlight, SmartScreen, SQL Server, Visual Basic, Visual C++, Visual Studio, Windows, the Windows logo, Windows Azure, Windows PowerShell, Windows Server, the Windows start button, and Windows Vista are registered trademarks or trademarks of Microsoft Corporation. Microsoft product screen shots are reprinted with permission from Microsoft Corporation.

All other trademarks, service marks, and company names in this document or website are properties of their respective owners.

The open source content used in Hitachi Vantara products may be found within the Product documentation or you may request a copy of such information (including source code and/or modifications to the extent the license for any open source requires Hitachi make it available) by sending an email to OSS_licensing@hitachivantara.com.

Feedback

Hitachi Vantara welcomes your feedback. Please share your thoughts by sending an email message to Docs-Feedback@hitachivantara.com. To assist the routing of this message, use the paper number in the subject and the title of this guide in the text.

Thank you!

Revision history

Changes	Date
▪ Initial release.	November 2025

Best Practices Guide

This document provides a quick and reliable procedure for storage backup and recovery using Thin Image Advanced (TIA) snapshots of a running PostgreSQL database on Red Hat Enterprise Linux (RHEL) systems. It is intended for system administrators, database administrators (DBAs), and storage engineers who need to capture storage array-based snapshots during runtime without interrupting database service. It details the steps involved, including online backup management with PostgreSQL backup control functions, temporary filesystem freezing through fsfreeze, and safe unfreezing after the snapshot is taken. This guide is especially valuable in environments using SAN or advanced storage replication technologies.

Accessing Product Downloads

Product software, drivers, and firmware downloads are available on Hitachi Vantara Support Connect: <https://support.hitachivantara.com/>.

Log in and select Product Downloads to access the most current downloads, including updates that may have been made after the release of the product

Getting help

[Hitachi Vantara Support Connect](https://support.hitachivantara.com/) is the destination for technical support of products and solutions sold by Hitachi Vantara. To contact technical support, log on to Hitachi Vantara Support Connect for contact information: https://support.hitachivantara.com/en_us/contact-us.html.

[Hitachi Vantara Community](https://community.hitachivantara.com/) is a global online community for customers, partners, independent software vendors, employees, and prospects. It is the destination to get answers, discover insights, and make connections. Join the conversation today! Go to community.hitachivantara.com, register, and complete your profile.

Value proposition

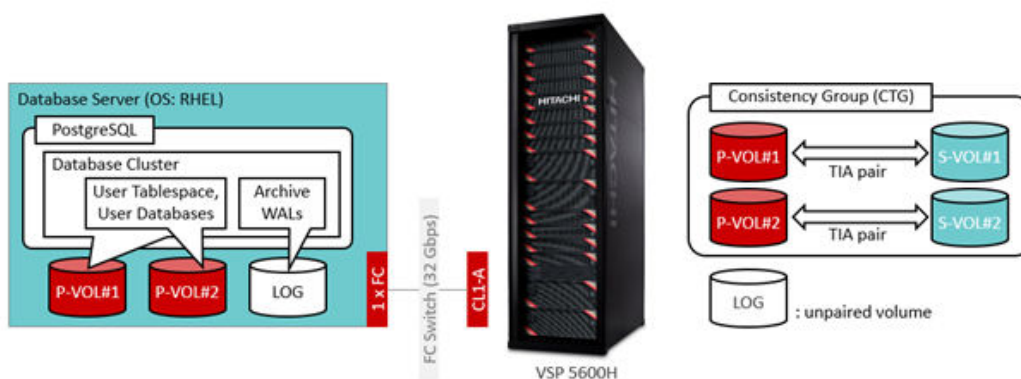
This solution offers the following benefits:

- Zero-downtime protection - Online backup management with PostgreSQL backup control functions makes it possible to back up the PostgreSQL database without downtime.
- Fast recovery - Restoring data from storage array-based snapshots enables fast restore and recovery.
- Hitachi VSP One Block and VSP 5000 series integration - Leverages enterprise-grade storage features (for example, Thin Image Advanced and CTG consistency) for high reliability and fast operations.

Introduction

This document outlines a standardized and repeatable procedure for storage backup and recovery using Thin Image Advanced snapshots of an active PostgreSQL database running on Linux (RHEL) systems, using Hitachi VSP One Block or Hitachi VSP 5000 series storage. It is possible to safely take online backups of the PostgreSQL database without disrupting live database operations, by coordinating online backup management with PostgreSQL backup control functions and Thin Image Advanced snapshots. In the event of a failure, coordinating with continuous archiving of the PostgreSQL database enables roll-forward recovery using archived Write-Ahead Logs (WAL), guaranteeing application-consistent recovery. This document provides an explanation of recovery scenarios for logical failures assumed in a PostgreSQL database. The guide helps database administrators, system engineers, and operations teams managing PostgreSQL in both production and testing environments.

The following illustration shows a block diagram of the key characteristics and components of the test environment.



Hardware requirements

For this validation, the following hardware configuration was used:

- Server Model: HA800 series
- CPU: Dual Intel Xeon Silver 4316 (20 cores each)
- Memory: 256 GB DDR4 RAM
- Storage: Hitachi VSP 5600H
- PostgreSQL Database Cluster Disk: 2 x 1 TB LUNs (OPEN-V), One of them is configured as a snapshot volume (SVOL)
- PostgreSQL User Tablespace, Databases Disk: 2 x 1 TB LUNs (OPEN-V), One of them is configured as a snapshot volume (SVOL)
- PostgreSQL Log Disk: 1 x 1 TB LUNs (OPEN-V)
- Filesystem: XFS

Software requirements

For this validation, the following software components were used:

- Operating system:
 - Red Hat Enterprise Linux (RHEL) 9.5
- Database software:
 - PostgreSQL 17.5
 - Installed using official repositories for PostgreSQL
- Utilities and tools:
 - `psql` command-line utility (part of the PostgreSQL installation)
 - `fsfreeze` utility and disk-related utility (part of `util-linux` package)
- Permissions:
 - Root access or `sudo` privileges to run `fsfreeze`, disk-related commands, or snapshot-related commands
 - PostgreSQL Server `postgres` or equivalent admin credentials to connect using `psql` or run PostgreSQL-related commands

Operational workflow

The following procedures outline best practices for using Thin Image Advanced for Active PostgreSQL Database.

Install PostgreSQL on the database server

Procedure

1. Create 5 DRS volumes that are 1 TB each on the Hitachi VSP 5600H storage system.
2. Map 3 volumes to the database server and perform LUN discovery. The remaining 2 volumes that were not mapped will serve as SVOLs for TIA.
3. Format 3 volumes with the XFS file system.

```
mkfs.xfs /dev/sdc
```

4. Create 3 directories to serve as mount points for each volume.
 - The directory to store the database cluster: `mkdir -p /PostgreSQL_cluster_5600`
 - The directory to store the user tablespace: `mkdir -p /PostgreSQL_data_5600`
 - The directory to store the archive WALs: `mkdir -p /PostgreSQL_log_5600`
5. Mount each of 3 volumes to a different directory.

```
mount /dev/sdc /PostgreSQL_cluster_5600
```

6. Change the ownership of 3 directories to the database user (postgres).

```
chown -R postgres:postgres /PostgreSQL*_5600
```

7. Use `initdb` to create a new PostgreSQL database cluster. Specify the directory to store the database cluster (one of the mount points created on the storage disk) with the `-D` option.

```
initdb -D /PostgreSQL_cluster_5600/data
```

8. Enable Continuous Archiving. Edit the `postgresql.conf` file and modify the `archive_mode` and `archive_command` settings. The changes will take effect by restarting the PostgreSQL server.

- `archive_mode = on`
- `archive_command = 'test ! -f /PostgreSQL_log_5600/archive/%f && cp %p /PostgreSQL_log_5600/archive/%f'`

9. Create a tablespace named `TIA_data`. Specify the directory to store the tablespace (one of the mount points created on the storage disk) in the `LOCATION` clause.

```
CREATE TABLESPACE TIA_data LOCATION '/PostgreSQL_data_5600';
```

10. Create a database named `TIA_test`, specifying the previously created tablespace.

```
CREATE DATABASE TIA_test TABLESPACE TIA_data;
```

11. Initiate TIA snapshot pair creation using the `raidcom add snapshot` command, specifying the `PVOL` (the volume that stores the database cluster), `SVOL`, `pool`, `snapshot group`, and `CTG` (consistency group) options.

```
raidcom add snapshot -ldev_id 00:04 00:05 -pool SQL-Pool-TIA -snapshotgroup postgresql -snap_mode cascade CTG
```

12. Initiate TIA snapshot pair creation for the second volume using the `raidcom add snapshot` command, specifying `PVOL` (the volume that stores the tablespace), `SVOL`, `pool`, `snapshot group`, and `CTG` (consistency group) options. Be sure to specify the same snapshot group name as the one used for the pair associated with the volume that stores the database cluster.

```
raidcom add snapshot -ldev_id 00:08 00:09 -pool SQL-Pool-TIA -snapshotgroup postgresql -snap_mode cascade CTG
```

13. Wait until all volumes reach the `PAIR` (Paired) state.

```
raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format_time
```

Create a backup

Procedure

1. Use a tool that repeatedly executes a large number of INSERT operations to run queries on the TIA_test database.
2. Connect to the PostgreSQL database server as a user with superuser privileges or with execution privileges for backup control functions, and prepare to begin an online backup using the `pg_backup_start` function. The session in which the `pg_backup_start` function was executed must be kept open and not closed.

```
SELECT pg_backup_start('label',true);
```

3. Using a different prompt, flush the operating system's file cache to disk.

```
sync; sync; sync
```

4. Freeze the XFS file systems mounted on the directories that store the database cluster and tablespace to prepare for a consistent snapshot.

```
fsfreeze -f /PostgreSQL_cluster_5600; fsfreeze -f /PostgreSQL_data_5600;
```

5. Split the snapshot using the `raidcom modify snapshot` command, specifying the snapshot group and `-snapshot_data split` option. Wait until all volumes are in the PSUS (Pair Suspended) state.

```
raidcom modify snapshot -snapshotgroup postgresql -snapshot_data split
```

```
raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format_time
```

6. After the snapshot is complete, unfreeze the XFS file system.

```
fsfreeze -u /PostgreSQL_cluster_5600; fsfreeze -u /PostgreSQL_data_5600;
```

7. Finish performing an online backup by using the `pg_backup_stop` function. Be sure to perform this operation in the same session that was kept open after executing the `pg_backup_start` function.
8. A backup of the files in the directory where archived WALs are output can be made by copying them to another location.

```
cp -a /PostgreSQL_log_5600/archive /PostgreSQL_log_5600/archive.`date "+%Y%m%d_%H%M%S" `
```

Next steps

If a failure occurs and recovery from a backup is required, the following procedure describes the restore and recovery operation.

Recovery from logical failures

The following steps provide an explanation of recovery scenarios for logical failures assumed in a PostgreSQL database.

Procedure

1. Stop the PostgreSQL database server, if it's running.

```
pg_ctl stop
```

2. Unmount the directories that store the database cluster and tablespace.

- `umount /PostgreSQL_cluster_5600`

- `umount /PostgreSQL_data_5600`

3. Restore the snapshot using the `raidcom modify snapshot` command, specifying the snapshot group and `-snapshot_data restore` option. Wait until all volumes return to the PSUS (Pair Suspend) state after going through RCPY (Reversed Copy).

- `raidcom modify snapshot -snapshotgroup postgresql -snapshot_data restore`

- `raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format_time`

4. Mount the directories that store the database cluster and tablespace.

- `mount /dev/sdc /PostgreSQL_cluster_5600`

- `mount /dev/sdb /PostgreSQL_data_5600`

5. Copy the archived WAL files that were backed up to another location into the directory where the archived WALs are output. If necessary, relocate any data remaining in the directory where archived WALs are output to another location beforehand.

```
cp -a /PostgreSQL_log_5600/archive.yyyymmdd_hhmmss/* /PostgreSQL_log_5600/
archive
```

6. Delete the `postmaster.pid` file in the restored database cluster.

```
rm -Rf /PostgreSQL_cluster_5600/data/postmaster.pid
```

- Configure the system to apply WAL when PostgreSQL is started in recovery mode. Edit the `postgresql.conf` file and modify the `restore_command` settings.

```
restore_command = 'cp "/PostgreSQL_log_5600/archive/%f" "%p"'
```

- Create a `recovery.signal` file in the database cluster to start PostgreSQL in recovery mode.

```
touch /PostgreSQL_cluster_5600/data/recovery.signal
```

- Start the PostgreSQL database server.

```
pg_ctl start
```

- Based on the archive, the WAL is applied up to the latest state, allowing operations that occurred after the restored TIA snapshot was taken to be rolled forward.

Results and observations

Procedure

- Install PostgreSQL on the database server.

```
[root@sql-rhel001 ~]# /usr/pgsql-17/bin/psql -V
psql (PostgreSQL) 17.5
[root@sql-rhel001 ~]#
```

- Create 5 DRS volumes that are 1 TB each on the Hitachi VSP 5600H storage system.

```
c:\HORCM\etc>raidcom add ldev -ldev_id 00:04 -drs -capacity_saving deduplication_compression -request_id auto -pool 0 -capacity 1024G
REQID : 1
c:\HORCM\etc>
```

```

c:VHORCM#etc>raidcom get ldev -ldev_id 00:04
Serial# : 590019
LDEV : 4
SL : 0
CL : 0
VOL_TYPE : OPEN-V-CVS
VOL_Capacity(BLK) : 2147483648
NUM_PORT : 0
PORTs :
F_POOLID : NONE
VOL_ATTR : CVS : HDP : DRS
CMP : -
EXP_SPACE : -
B_POOLID : 0
LDEV_NAMING :
STS : NML
D_PM : -
D_PM(%) : -
OPE_TYPE : NONE
OPE_RATE : 100
MP# : 0
ASSIGNED_MP# : 0
SSID : 0004
Used_Block(BLK) : 0
FLA(MB) : Disable
RSV(MB) : 0
CSV_Status : ENABLED
CSV_PROGRESS(%) : -
CSV_Mode : DEDUP+COMPRESS
COMPRESSION_ACCELERATION : ENABLED
COMPRESSION_ACCELERATION_STATUS : ENABLED
CSV_PROCESS_MODE : INLINE
DEDUPLICATION_DATA : ENABLED
ALUA : Disable
RSGID : 0
PWSV_S : -
CL_MTG : N
c:VHORCM#etc>_

```

3. Format 3 volumes with the XFS file system.
4. Create 3 directories to serve as mount points for each volume.
5. Mount each of 3 volumes to a different directory.
6. Change the ownership of 3 directories to the database user (postgres).

```

[root@sql-rhel001 ~]# mkfs.xfs /dev/sdc
meta-data=/dev/sdc          isize=512    agcount=4, agsize=67108864 blks
           =                sectsz=512    attr=2, projid32bit=1
           =                crc=1        finobt=1, sparse=1, rmapbt=0
           =                reflink=1    bigtime=1 inobtcount=1 nnext64=0
data      =                bsize=4096   blocks=268435456, imaxpct=5
           =                sunit=0     swidth=0 blks
naming    =version 2        bsize=4096   ascii-ci=0, ftype=1
log       =internal log    bsize=4096   blocks=131072, version=2
           =                sectsz=512   sunit=0 blks, lazy-count=1
realtime  =none           extsz=4096   blocks=0, rtextents=0
[root@sql-rhel001 ~]#
[root@sql-rhel001 ~]# mkdir -p /PostgreSQL_cluster_5600
[root@sql-rhel001 ~]#
[root@sql-rhel001 ~]# mount /dev/sdc /PostgreSQL_cluster_5600
[root@sql-rhel001 ~]#
[root@sql-rhel001 ~]# chown -R postgres:postgres /PostgreSQL_cluster_5600
[root@sql-rhel001 ~]#

```

7. Use `initdb` to create a new PostgreSQL database cluster. Specify the directory to store the database cluster (one of the mount points created on the storage disk) with the `-D` option.

```

[postgres@sql-rhel001 ~]$ initdb -D /PostgreSQL_cluster_5600/data
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.UTF-8".
The default database encoding has accordingly been set to "UTF8".
The default text search configuration will be set to "english".

Data page checksums are disabled.

creating directory /PostgreSQL_cluster_5600/data ... ok
creating subdirectories ... ok
selecting dynamic shared memory implementation ... posix
selecting default "max_connections" ... 100
selecting default "shared_buffers" ... 128MB
selecting default time zone ... Asia/Tokyo
creating configuration files ... ok
running bootstrap script ... ok
performing post-bootstrap initialization ... ok
syncing data to disk ... ok

initdb: warning: enabling "trust" authentication for local connections
initdb: hint: You can change this by editing pg_hba.conf or using the option -A, or --auth-local and --auth-host, the next time you run initdb.

Success. You can now start the database server using:

    pg_ctl -D /PostgreSQL_cluster_5600/data -l logfile start

[postgres@sql-rhel001 ~]$

```

8. Enable Continuous Archiving. Edit the `postgresql.conf` file and modify the `archive_mode` and `archive_command` settings. The changes will take effect by restarting the PostgreSQL server.

```

[root@sql-rhel001 ~]# cd $PGDATA
[root@sql-rhel001 data]#
[root@sql-rhel001 data]# ls -al
total 68
drwx----- 19 postgres postgres 4096 Aug  5 07:31 .
drwx-----  3 postgres postgres  18 Aug  5 07:04 ..
drwx-----  5 postgres postgres  33 Aug  5 07:04 base
-rw-----  1 postgres postgres  30 Aug  5 07:17 current_logfiles
drwx-----  2 postgres postgres 4096 Aug  5 07:22 global
drwx-----  2 postgres postgres  32 Aug  5 07:17 log
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_commit_ts
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_dynshmem
-rw-----  1 postgres postgres 5711 Aug  5 07:04 pg_hba.conf
-rw-----  1 postgres postgres 2640 Aug  5 07:04 pg_ident.conf
drwx-----  4 postgres postgres   68 Aug  5 07:31 pg_logical
drwx-----  4 postgres postgres  36 Aug  5 07:04 pg_multixact
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_notify
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_replslot
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_serial
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_snapshots
drwx-----  2 postgres postgres  25 Aug  5 07:31 pg_stat
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_stat_tmp
drwx-----  2 postgres postgres  18 Aug  5 07:04 pg_subtrans
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_tblspc
drwx-----  2 postgres postgres   6 Aug  5 07:04 pg_twophase
-rw-----  1 postgres postgres   3 Aug  5 07:04 PG_VERSION
lrwxrwxrwx  1 postgres postgres  24 Aug  5 07:04 pg_wal -> /PostgreSQL_log_5600/
drwx-----  2 postgres postgres  18 Aug  5 07:04 pg_xact
-rw-----  1 postgres postgres  88 Aug  5 07:04 postgresql.auto.conf
-rw-----  1 postgres postgres 30694 Aug  5 07:04 postgresql.conf
-rw-----  1 postgres postgres  64 Aug  5 07:17 postmaster.opts
[root@sql-rhel001 data]#
[root@sql-rhel001 data]#
[root@sql-rhel001 data]# vi postgresql.conf

# - Archiving -
archive_mode = on                # enables archiving; off, on, or always
                                # (change requires restart)
#archive_library = ''            # library to use to archive a WAL file
                                # (empty string indicates archive_command should
                                # be used)
archive_command = 'test ! -f /PostgreSQL_log_5600/archive/%f && cp %p /PostgreSQL_log_5600/archive/%f'
                                # placeholders: %p = path of file to archive
                                #           %f = file name only
                                # e.g. 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'
#archive_timeout = 0            # force a WAL file switch after this
                                # number of seconds; 0 disables

```

```

postgres@sql-rhel001 ~]$ pg_ctl restart
waiting for server to shut down.... done
server stopped
waiting for server to start....2025-08-05 07:49:54.382 JST [14360] LOG:  redirecting log output to logging collector process
025-08-05 07:49:54.382 JST [14360] HINT:  Future log output will appear in directory "log".
done
server started
postgres@sql-rhel001 ~]$
postgres@sql-rhel001 ~]$ pg_ctl status
pg_ctl: server is running (PID: 14360)
usr/pgsql-17/bin/postgres "-D" "/PostgreSQL_cluster_5600/data"
postgres@sql-rhel001 ~]$

```

9. Create a tablespace named `TIA_data`. Specify the directory to store the tablespace (one of the mount points created on the storage disk) in the `LOCATION` clause.

```

postgres=# CREATE TABLESPACE TIA_data LOCATION '/PostgreSQL_data_5600';
CREATE TABLESPACE
postgres=#
postgres=# \db

```

List of tablespaces		
Name	Owner	Location
pg_default	postgres	
pg_global	postgres	
tia_data	postgres	/PostgreSQL_data_5600

```

(3 rows)
postgres=#

```

10. Create a database named `TIA_test`, specifying the previously created tablespace.

```

postgres=# CREATE DATABASE TIA_test TABLESPACE TIA_data;
CREATE DATABASE
postgres=#
postgres=# \l

```

List of databases								
Name	Owner	Encoding	Locale Provider	Collate	Ctype	Locale	ICU Rules	Access privileges
postgres	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			
template0	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=c/postgres +
template1	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=c/postgres +
tia_test	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			postgres=CtC/postgres

```

(4 rows)
postgres=#

```

11. Initiate TIA snapshot pair creation using the `raidcom add snapshot` command, specifying the `PVOL` (the volume that stores the database cluster), `SVOL`, `pool`, `snapshot group`, and `CTG` (consistency group) options.

```

c:\HORCM\etc>raidcom add snapshot -ldev_id 00:04 00:05 -pool SQL-Pool-TIA -snapshotgroup postgresql -snap_mode cascade CTG
c:\HORCM\etc>

```

12. Initiate TIA snapshot pair creation for the second volume using the `raidcom add snapshot` command, specifying `PVOL` (the volume that stores the tablespace), `SVOL`, `pool`, `snapshot group`, and `CTG` (consistency group) options. Be sure to specify the same snapshot group name as the one used for the pair associated with the volume that stores the database cluster.

```

c:\HORCM\etc>raidcom add snapshot -ldev_id 00:08 00:09 -pool SQL-Pool-TIA -snapshotgroup postgresql -snap_mode cascade CTG
c:\HORCM\etc>

```

13. Wait until all volumes reach the `PAIR` (Paired) state.

```

c:\HORCM\etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
SnapShot_name P/S STAT Serial# LDEV# MU# P-LDEV# PID % MODE SPLT-TIME SLU C_LDEV# P R D_STAT
postgresql P-VOL PAIR 590019 4 3 5 0 - G-AR----- - N - N D PAIR
postgresql P-VOL PAIR 590019 8 3 9 0 - G-AR----- - N - N D PAIR
c:\HORCM\etc>

```

Verify the backup operation

The following describes the procedure for the backup operation.

Procedure

1. Use a tool that repeatedly executes a large number of INSERT operations to run queries on the TIA_test database.

```
[postgres@sql-rhel001 sql1]$ psql -U postgres -d tia_test -f burst.sql
CREATE TABLE
CREATE TABLE
CREATE TABLE
psql:burst.sql:54: NOTICE:  complete. Waiting for next cycle...
psql:burst.sql:54: NOTICE:  complete. Waiting for next cycle...
psql:burst.sql:54: NOTICE:  complete. Waiting for next cycle...
```

2. Connect to the PostgreSQL database server as a user with superuser privileges or with execution privileges for backup control functions, and prepare to begin an online backup using the `pg_backup_start` function. The session in which the `pg_backup_start` function was executed must be kept open and not closed.

```
postgres=# SELECT pg_backup_start('label',true);
pg_backup_start
-----
2/4C008C10
(1 row)
postgres=#
```

3. Using a different prompt, flush the operating system's file cache to disk.

```
[root@sql-rhel001 archive]# sync; sync; sync
[root@sql-rhel001 archive]#
```

4. Freeze the XFS file systems mounted on the directories that store the database cluster and tablespaces to prepare for a consistent snapshot.

```
[root@sql-rhel001 archive]# fsfreeze -f /PostgreSQL_cluster_5600; fsfreeze -f /PostgreSQL_data_5600;
[root@sql-rhel001 archive]#
```

5. Split the snapshot using the `raidcom modify snapshot` command, specifying the snapshot group and `-snapshot_data split` option. Wait until all volumes are in the PSUS (Pair Suspended) state.

```
c:\VHORCM\etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
Snapshot_name  P/S  STAT  Serial#  LDEV#  MU#  P-LDEV#  PID  %  MODE  SPLT-TIME  SLU  C_LDEV#  P  R  D  STAT
postgresql     P-VOL PAIR  590019  4  3  5  0  -  G--AR-----  -  -  N  D  PAIR
postgresql     P-VOL PAIR  590019  8  3  9  0  -  G--AR-----  -  -  N  D  PAIR

c:\VHORCM\etc>
c:\VHORCM\etc>raidcom modify snapshot -snapshotgroup postgresql -snapshot_data split

c:\VHORCM\etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
Snapshot_name  P/S  STAT  Serial#  LDEV#  MU#  P-LDEV#  PID  %  MODE  SPLT-TIME  SLU  C_LDEV#  P  R  D  STAT
postgresql     P-VOL PSUP  590019  4  3  5  0  81  G--AR-----  2025-09-29T17:07:02  N  -  N  D  PSUP
postgresql     P-VOL PSUP  590019  8  3  9  0  81  G--AR-----  2025-09-29T17:07:02  N  -  N  D  PSUP

c:\VHORCM\etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
Snapshot_name  P/S  STAT  Serial#  LDEV#  MU#  P-LDEV#  PID  %  MODE  SPLT-TIME  SLU  C_LDEV#  P  R  D  STAT
postgresql     P-VOL PSUS  590019  4  3  5  0  -  G--AR-----  2025-09-29T17:07:02  N  -  N  D  PSUS
postgresql     P-VOL PSUS  590019  8  3  9  0  -  G--AR-----  2025-09-29T17:07:02  N  -  N  D  PSUS

c:\VHORCM\etc>
```

6. After the snapshot is complete, unfreeze the XFS file system.

```
[root@sql-rhel001 archive]# fsfreeze -u /PostgreSQL_cluster_5600; fsfreeze -u /PostgreSQL_data_5600;
[root@sql-rhel001 archive]#
```

7. Finish performing an online backup by using the `pg_backup_stop` function. Be sure to perform this operation in the same session that was kept open after executing the `pg_backup_start` function.

```

postgres=# SELECT pg_backup_stop();
NOTICE:  all required WAL segments have been archived
          pg_backup_stop
-----
(2/53839108,"START WAL LOCATION: 2/4C008C10 (file 000000020000000200000013)+
CHECKPOINT LOCATION: 2/4C04C7F8
BACKUP METHOD: streamed
BACKUP FROM: primary
START TIME: 2025-08-29 08:21:01 JST
LABEL: label
START TIMELINE: 2
", "16388 /PostgreSQL_data_5600
")
(1 row)

postgres=#

```

8. A backup of the files in the directory where archived WALs are output can be made by copying them to another location.

```

[root@sql-rhel001 PostgreSQL_log_5600]# cp -a /PostgreSQL_log_5600/archive /PostgreSQL_log_5600/archive.`date +%Y%m%d_%H%M%S`
[root@sql-rhel001 PostgreSQL_log_5600]#
[root@sql-rhel001 PostgreSQL_log_5600]# ls -al
total 156
drwx----- 2 postgres postgres 8192 Sep  4 03:15 archive
drwx----- 2 postgres postgres 8192 Aug 29 08:28 archive.20250829_082854
[root@sql-rhel001 PostgreSQL_log_5600]#

```

Restore and recovery operation

If a failure occurs and recovery from a backup is required, the following describes the procedure for the restore and recovery operation to be performed. The following steps provide an explanation of recovery scenarios for logical failures assumed in a PostgreSQL database.

Procedure

1. Stop the PostgreSQL database server, if it's running.

```

[postgres@sql-rhel001 ~]$ pg_ctl stop
waiting for server to shut down... done
server stopped
[postgres@sql-rhel001 ~]$

```

2. Unmount the directories that store the database cluster and tablespace.

```

[root@sql-rhel001 archive-restore6-beforepg_wal]# umount /PostgreSQL_cluster_5600
[root@sql-rhel001 archive-restore6-beforepg_wal]# umount /PostgreSQL_data_5600
[root@sql-rhel001 archive-restore6-beforepg_wal]#
[root@sql-rhel001 archive-restore6-beforepg_wal]# df -h
Filesystem              Size  Used Avail Use% Mounted on
devtmpfs                 4.0M   0  4.0M   0% /dev
tmpfs                    252G   0  252G   0% /dev/shm
tmpfs                    101G  339M  101G   1% /run
efivarfs                  496K  360K  132K  74% /sys/firmware/efi/efivars
/dev/mapper/rhel-root    70G   12G   59G  17% /
/dev/sda2                960M  373M  588M  39% /boot
/dev/mapper/rhel-home   1.4T   9.9G  1.4T   1% /home
/dev/sda1                 599M   7.1M  592M   2% /boot/efi
tmpfs                     51G   52K   51G   1% /run/user/42
tmpfs                     51G   36K   51G   1% /run/user/0
/dev/sdd                  1.5T   67G  1.5T   5% /PostgreSQL_log_5600
[root@sql-rhel001 archive-restore6-beforepg_wal]#

```

- Restore the snapshot using the raidcom modify snapshot command, specifying the snapshot group and `-snapshot_data restore` option. Wait until all volumes return to the PSUS (Pair Suspend) state after going through RCPY (Reversed Copy).

```
c:#HORCM#etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
SnapShot_name      P/S  STAT  Serial# LDEV#  MU#  P-LDEV#  PID  % MODE      SPLT-TIME      SLU  C_LDEV#  P  R  D  STAT
postgresql         P-VOL PSUS  590019  4  3  5  0  - G--AR----- 2025-08-29T17:07:02 N  -  N  D  PSUS
postgresql         P-VOL PSUS  590019  8  3  9  0  - G--AR----- 2025-08-29T17:07:02 N  -  N  D  PSUS

c:#HORCM#etc>
c:#HORCM#etc>raidcom modify snapshot -snapshotgroup postgresql -snapshot_data restore

c:#HORCM#etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
SnapShot_name      P/S  STAT  Serial# LDEV#  MU#  P-LDEV#  PID  % MODE      SPLT-TIME      SLU  C_LDEV#  P  R  D  STAT
postgresql         P-VOL RCPY 590019  4  3  5  0  58 G--AR----- - - - - -
postgresql         P-VOL RCPY 590019  8  3  9  0  58 G--AR----- - - - - -

c:#HORCM#etc>raidcom get snapshot -snapshotgroup postgresql -fx -key detail -format time
SnapShot_name      P/S  STAT  Serial# LDEV#  MU#  P-LDEV#  PID  % MODE      SPLT-TIME      SLU  C_LDEV#  P  R  D  STAT
postgresql         P-VOL PSUS  590019  4  3  5  0  - G--AR----- 2025-08-29T17:07:02 N  -  N  D  PSUS
postgresql         P-VOL PSUS  590019  8  3  9  0  - G--AR----- 2025-08-29T17:07:02 N  -  N  D  PSUS

c:#HORCM#etc>
```

- Mount the directories that store the database cluster and tablespace.

```
[root@sql-rhel001 archive-restore6-beforepg_wal]# mount /dev/sdc /PostgreSQL_cluster_5600
[root@sql-rhel001 archive-restore6-beforepg_wal]# mount /dev/sdb /PostgreSQL_data_5600
[root@sql-rhel001 archive-restore6-beforepg_wal]#
[root@sql-rhel001 archive-restore6-beforepg_wal]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        4.0M   0  4.0M   0% /dev
tmpfs           252G   0  252G   0% /dev/shm
tmpfs           101G  339M  101G   1% /run
efivarfs       496K  360K  132K  74% /sys/firmware/efi/efivars
/dev/mapper/rhel-root 70G  12G  59G  17% /
/dev/sda2       960M  373M  588M  39% /boot
/dev/mapper/rhel-home 1.4T  9.9G  1.4T   1% /home
/dev/sda1       599M   7.1M  592M   2% /boot/efi
tmpfs           51G   52K   51G   1% /run/user/42
tmpfs           51G   36K   51G   1% /run/user/0
/dev/sdd        1.5T   67G  1.5T   5% /PostgreSQL_log_5600
/dev/sdc        1.0T  8.3G 1016G   1% /PostgreSQL_cluster_5600
/dev/sdb        1.0T  13G 1011G   2% /PostgreSQL_data_5600
[root@sql-rhel001 archive-restore6-beforepg_wal]#
```

- Copy the archived WAL files that were backed up to another location into the directory where the archived WALs are output. If necessary, relocate any data remaining in the directory where archived WALs are output to another location beforehand.

```
root@sql-rhel001 PostgreSQL_log_5600]# cp -a /PostgreSQL_log_5600/archive.20250829_082854/* archive
root@sql-rhel001 PostgreSQL_log_5600]#
```

- Delete the `postmaster.pid` file in the restored database cluster.

```
[postgres@sql-rhel001 data]$ rm -rf postmaster.pid
[postgres@sql-rhel001 data]$
```

- Configure the system to apply WAL when PostgreSQL is started in recovery mode. Edit the `postgresql.conf` file and modify the `restore_command` settings.

```
# - Archive Recovery -

# These are only used in recovery mode.

restore_command = 'cp /PostgreSQL_log_5600/archive/%f %p'
# placeholders: %p = path of file to restore
#               %f = file name only
# e.g. 'cp /mnt/server/archivedir/%f %p'

#archive_cleanup_command = '' # command to execute at every restartpoint
#recovery_end_command = ''   # command to execute at completion of recovery
```

- Create a `recovery.signal` file in the database cluster to start PostgreSQL in recovery mode.

```
[postgres@sql-rhel001 data]$ touch /PostgreSQL_cluster_5600/data/recovery.signal
[postgres@sql-rhel001 data]$
```

- Start the PostgreSQL database server.

```
[postgres@sql-rhel001 ~]$ pg_ctl start
waiting for server to start....2025-09-01 07:17:27.044 JST [1793137] LOG: redirecting log output to logging collector process
2025-09-01 07:17:27.044 JST [1793137] HINT: Future log output will appear in directory "log".
done
server started
[postgres@sql-rhel001 ~]$
```

10. Based on the archive, the WAL is applied up to the latest state, allowing operations that occurred after the restored TIA snapshot was taken to be rolled forward.

Conclusion

This guide outlines a structured and repeatable procedure for storage backup and recovery of an actively running PostgreSQL database on RHEL, using Hitachi VSP One Block or Hitachi VSP 5000 series storage. By following the outlined steps, combining online backup management for PostgreSQL databases, freezing the XFS filesystem, and managing storage array-based snapshot operations with Hitachi Command Control Interface (CCI), administrators can confidently capture reliable and safe backups of live databases without downtime.

Additionally, restoring Thin Image Advanced snapshot data and archived WAL files that were backed up through coordination with continuous archiving of the PostgreSQL database, roll-forward processing using WAL is performed by the archive recovery at PostgreSQL startup. This enables the data to be restored to a state that maintains application consistency and ensures database integrity.

These backup and recovery procedures achieve data protection, ensuring minimal impact on production workloads.

References

See [PostgreSQL Installation on RHEL Linux](#) for information about installing and configuring PostgreSQL on Red Hat Enterprise Linux.

Hitachi Vantara



Corporate Headquarters
2535 Augustine Drive
Santa Clara, CA 95054 USA

HitachiVantara.com/contact