



Using TMF COMMITHOLDMODE for RDF/ZLT

A Technical Overview

Table of contents

Abstract.....	3
Introduction.....	3
Intended Audience.....	3
Recommended Pre-Reading.....	3
Understanding ZLT configurations.....	3
Definition from RDF Manual.....	4
RDF/ZLT Configurations.....	4
Understanding terminology.....	6
COMMITHOLDMODE.....	6
Configuration Syntax.....	6
Purpose of the Attribute.....	6
COMMITHOLDTIMER.....	7
Configuration Syntax.....	7
Purpose of the Attribute.....	8
Further Reading.....	8
Disaster Management Steps with COMMITHOLDMODE.....	8
Step 1 – RDF/ZLT Configured.....	8
Step 2 – DP2 Recognizes Loss of the Remote Mirror.....	9
Step 3 – DP2 TMF Communication and Timer Popup.....	10
Step 4 – End of Timer.....	11
TMF Crash (Diagram continued after DP2 Communicates to TMF).....	11
Suspension of ZLT Protection (Diagram continued after DP2 Communicates to TMF).....	12
Technical Information.....	13
Subsystems Involved.....	13
How Applications Perceive COMMITHOLD ON Functionality.....	13
Some Permutations and Combinations.....	13
Successful write to both mirrors with COMMITHOLD OFF.....	14
Successful write to only one mirror with COMMITHOLD OFF.....	14
Successful write to both mirrors with COMMITHOLD ON.....	14
Problem in writing to the local mirror with COMMITHOLD ON.....	14
Problem in writing to the REMOTE mirror with COMMITHOLD ON.....	14
Performance Discussion.....	15
Issues with Reducing Timeout Value for COMMITHOLDMODE.....	15
Why We Cannot Support Only COMMITHOLD ON with No TIMER.....	15
Finally, At What Level Is COMMITHOLD Applied?.....	15
For more information.....	17

Abstract

This whitepaper offers an overview of some technical aspects of the COMMITHOLDMODE attribute of the TMF audit trail with reference to NonStop Remote Database Facility (RDF) software to achieve its unique Zero Lost Transaction (ZLT) option. It provides important details about the interaction between TMF, DP2, and RDF Subsystems regarding COMMITHOLDMODE and successful RDF recovery.

Other effects of this attribute on user applications and some important performance implications are also discussed.

Introduction

Two key elements of the HP NonStop Transaction Management Facility (TMF) software have significant impact on the success of the RDF/ZLT solution:

1. COMMITHOLDMODE
2. COMMITHOLDTIMER

This document provides an in-depth analysis of these two attributes and their relevance in the way RDF/ZLT software operates.

Intended Audience

This document summarizes ideas exchanged by members of the HP NonStop RDF, TMF, and DP2 development teams, prompted by customer inquiries. Those who will benefit most from this summary are:

1. Customers wanting to use RDF/ZLT with TMF COMMITHOLDMODE
2. System managers administering RDF/ZLT
3. NonStop system programmers interested in learning about RDF/ZLT with TMF COMMITHOLDMODE

Recommended Pre-Reading

To get the most benefit of this review, you should be familiar with these documents/chapters:

- HP NonStop Transaction Management Facility (TMF) Introduction - <http://docs.hp.com/en/522414-001/522414-001.pdf> - Chapter 1 - TMF Overview
- HP NonStop TMF Operations and Recovery Guide - <http://docs.hp.com/en/522417-002/522417-002.pdf> - Chapter 1, 2 - Overview of TMF Maintenance and Recovery, Routine Maintenance
- HP NonStop TMF Reference Manual - <http://docs.hp.com/en/522418-003/522418-003.pdf> - Chapters 2, 3 - Using TMFCOM, TMFCOM command Set

Understanding ZLT configurations

As background information, it is important to understand RDF/ZLT basics.

Definition from RDF Manual

Zero Lost Transactions (ZLT), functionality available only with the RDF/ZLT product, ensures that no transactions that commit on the primary system are lost on the RDF backup system if the primary system is downed by an unplanned outage. RDF achieves this through the use of remote mirroring for the relevant TMF audit trail volume(s). This means that one mirror of an audit trail volume remains local to the primary system, but the other mirror is located at a remote standby site.

When a primary system is downed by some unplanned outage or disaster, there may be some audit data that the RDF extractor on the primary system was unable to send to the backup system before the outage. With ZLT functionality, RDF fetches all remaining audit data from the remote mirror, thereby guaranteeing no loss of committed data during the RDF takeover operation.

However, if a remote mirror is not available at the time of the outage, ZLT functionality cannot be guaranteed. ZLT functionality will succeed only if you enable the TMF COMMITHOLDMODE capability on your primary system by including the COMMITHOLDMODE parameter in a TMFCOM ALTER AUDITTRAIL command. When COMMITHOLDMODE is enabled and a remote mirror fails, TMF suspends all commit operations.

RDF/ZLT Configurations

Figure 1: ZLT configuration with single standby/backup system with remote mirror audit disk at backup

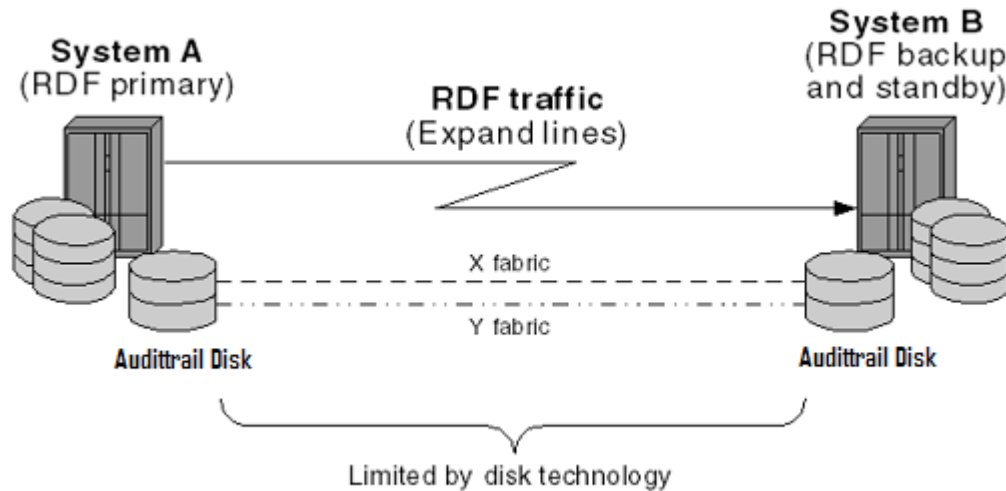


Figure 2: ZLT configuration with single standby/backup system with remote mirror at intermediate site

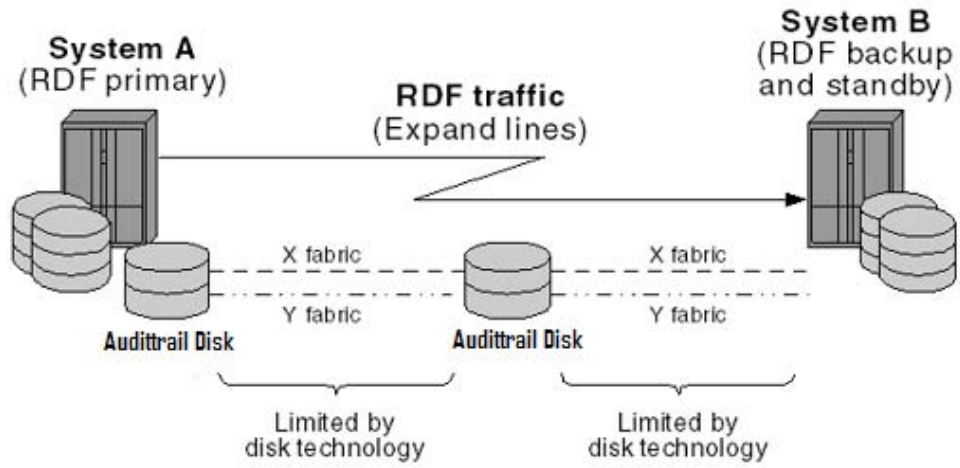
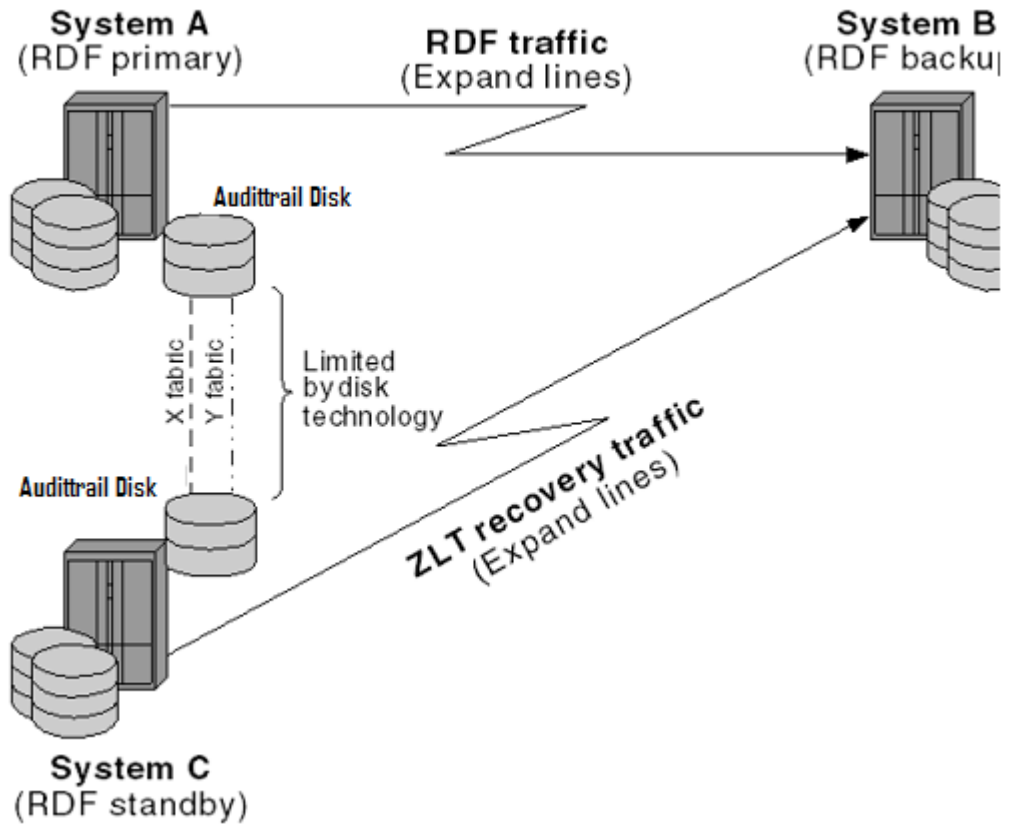


Figure 3: ZLT configuration with standby and backups at separate sites



Understanding terminology

COMMITHOLDMODE

COMMITHOLDMODE (referred to as COMMITHOLD) is an attribute available for AUDITTRAIL (AT) management on systems implementing RDF/ZLT.

Configuration Syntax

Figure 4: COMMITHOLDMODE is available with the TMFCOM ALTER AUDITTRAIL command

```
ALTER AUDITTRAIL {MASTER | MAT }
                 {AUX [ILIARY] nn}

[,AUDITDUMP {ON | OFF}]

[,ADDACTIVEVOL[S] {volume
                  {(volume [,volume]...) }]}]
[,DELETEACTIVEVOL[S] {volume
                     {(volume [,volume]...) }]}]

[,FILESPERVOLUME integer]
[,RESET FILESPERVOLUME ]

[,ADDOVERFLOWVOL[S] {volume
                    {(volume [,volume]...) }]}]
[,DELETEOVERFLOWVOL[S] {volume
                       {(volume [,volume]...) }]}]

[,OVERFLOWTHRESHOLD integer]
[,RESET OVERFLOWTHRESHOLD ]

[,ADDRESTOREVOL[S] {volume
                   {(volume [,volume]...) }]}]
[,DELETERESTOREVOL[S] {volume
                      {(volume [,volume]...) }]}]

[,MAXRETAINEDATFILES integer]
[,RESET MAXRETAINEDATFILES ]

[,BEGINTRANSDISABLE integer]
[,RESET BEGINTRANSDISABLE ]

[,FILESIZE integer]
[,RESET FILESIZE ]

[,COMMITHOLDMODE {ON | OFF | SUSPEND}]
[,RESET COMMITHOLDMODE]

[,COMMITHOLDTIMER
 {timeout [ON TIMEOUT {SUSPEND | CRASH}] | -1}]
```

Purpose of the Attribute

You must set the COMMITHOLD attribute "ON" to ensure that TMF on the primary RDF system will "hold" COMMIT operations for a preset period of time (discussed later) when a hardware or communication failure renders the remote mirror AT inaccessible. After reaching the pre-set time, TMF can be preconfigured to do one of the following:

1. Suspend COMMIT operations until instructed to restart
2. Crash the TMF Subsystem on the RDF Primary System

COMMITHOLDTIMER

COMMITHOLDTIMER (or CHTIMER) is an attribute used in conjunction with the COMMITHOLD attribute for AT on systems implementing RDF/ZLT.

Configuration Syntax

Figure 5: COMMITHOLDTIMER is available with the TMFCOM ALTER AUDITTRAIL command

```
ALTER AUDITTRAIL {MASTER | MAT }
                 {AUX [ILIARY] nn}

[,AUDITDUMP {ON | OFF}]

[,ADDACTIVEVOL[S] {volume
                  {(volume [,volume]...) }]}
[,DELETEACTIVEVOL[S] {volume
                     {(volume [,volume]...) }]}

[,FILESPELVOLUME integer]
[,RESET FILESPERVOLUME ]

[,ADDOVERFLOWVOL[S] {volume
                    {(volume [,volume]...) }]}
[,DELETEOVERFLOWVOL[S] {volume
                        {(volume [,volume]...) }]}

[,OVERFLOWTHRESHOLD integer]
[,RESET OVERFLOWTHRESHOLD ]

[,ADDRESTOREVOL[S] {volume
                   {(volume [,volume]...) }]}
[,DELETERESTOREVOL[S] {volume
                       {(volume [,volume]...) }]}

[,MAXRETAINEDATFILES integer]
[,RESET MAXRETAINEDATFILES ]

[,BEGINTRANSDISABLE integer]
[,RESET BEGINTRANSDISABLE ]

[,FILESIZE integer]
[,RESET FILESIZE ]

[,COMMITHOLDMODE {ON | OFF | SUSPEND}]
[,RESET COMMITHOLDMODE]

[,COMMITHOLDTIMER
 {timeout [ON TIMEOUT {SUSPEND | CRASH}] | -1}]
```

Purpose of the Attribute

The CHTIMER attribute sets the duration for which TMF, along with the DP2 subsystem (DP2), determines if it needs to declare the remote mirror AT down, and take the required subsequent action, as set by the user. The preconfigured actions can be:

- Suspend COMMIT operations on the RDF primary system until instructed to restart
- Crash the TMF Subsystem on RDF primary system

Further Reading

For more details, see the HP NonStop TMF Reference Manual:

1. G Series - <http://docs.hp.com/en/522418-003/522418-003.pdf>
2. H Series - <http://docs.hp.com/en/540138-001/540138-001.pdf>

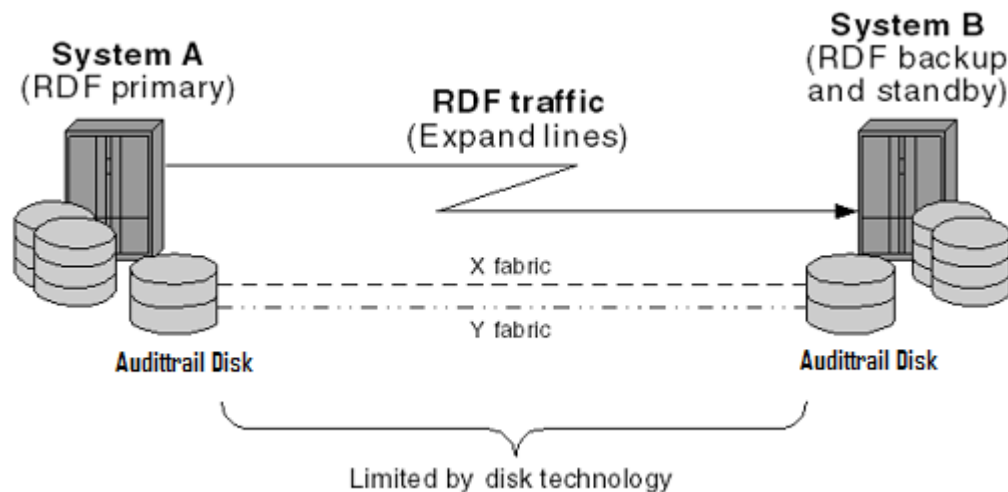
Disaster Management Steps with COMMITHOLDMODE

It is very important to ensure that all operations related to COMMITHOLDMODE be carried out only when the attribute is set to “ON” in the TMF subsystem for an AT and RDF/ZLT is implemented. For details on using this command refer to Figure 5 on page 8.

Following are the sequence of steps triggered when COMMITHOLDMODE is **ON** and when a hardware or communication failure renders the remote mirror AT inaccessible.

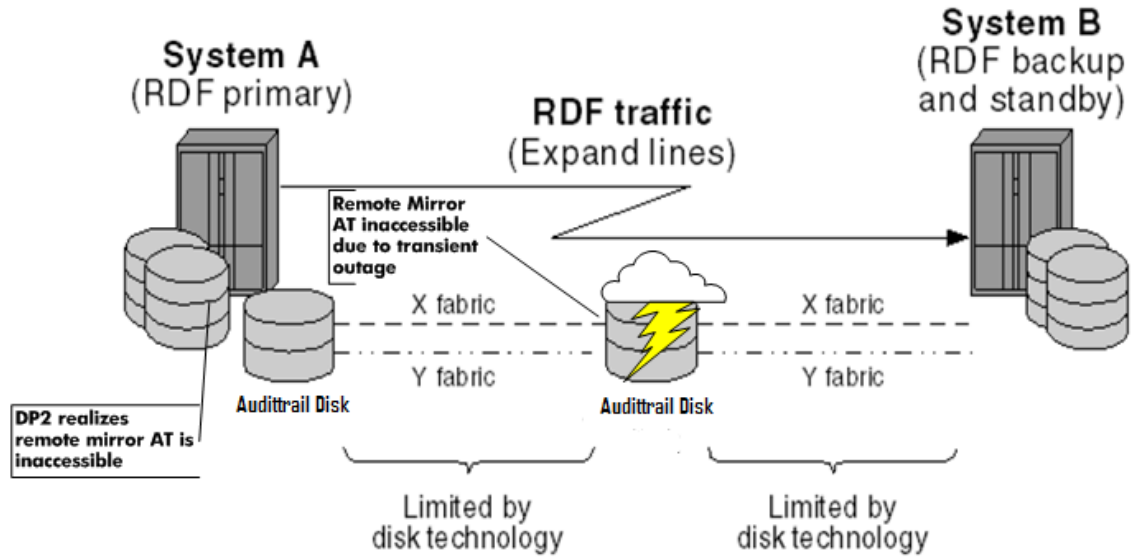
Step 1 – RDF/ZLT Configured

Figure 6: Basic RDF/ZLT environment with optional intermediate location of audit mirror



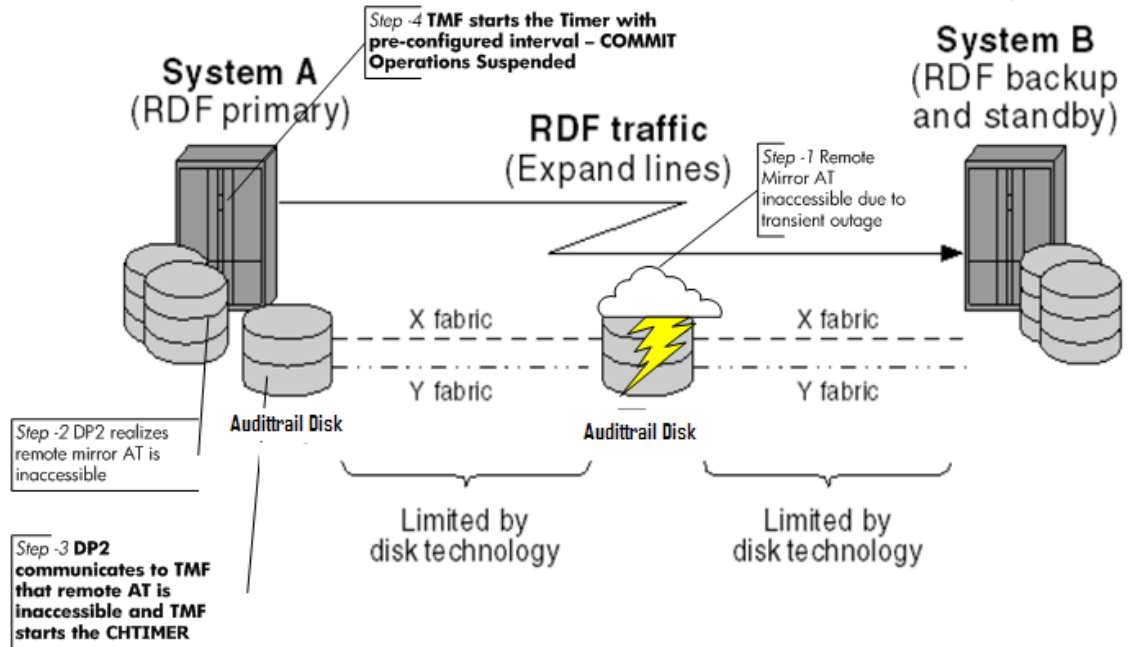
Step 2 – DP2 Recognizes Loss of the Remote Mirror

Figure 7: DP2 recognizes the loss of audit mirror in an outage



Step 3 – DP2 TMF Communication and Timer Popup

Figure 8: DP2 informs TMF and TMF starts CHTIMER

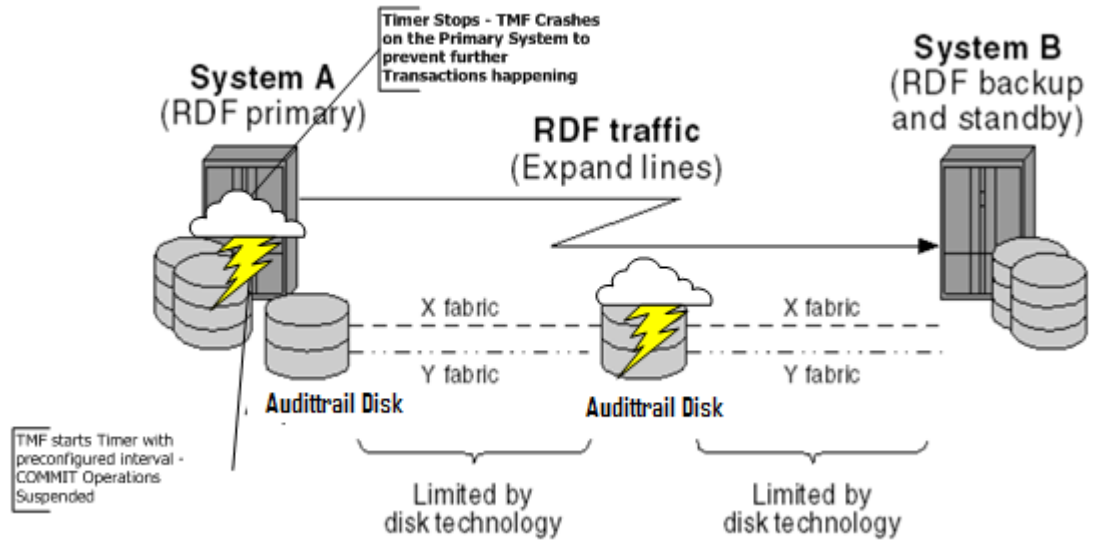


Step 4 – End of Timer

There are two possible outcomes based on the user configuration:

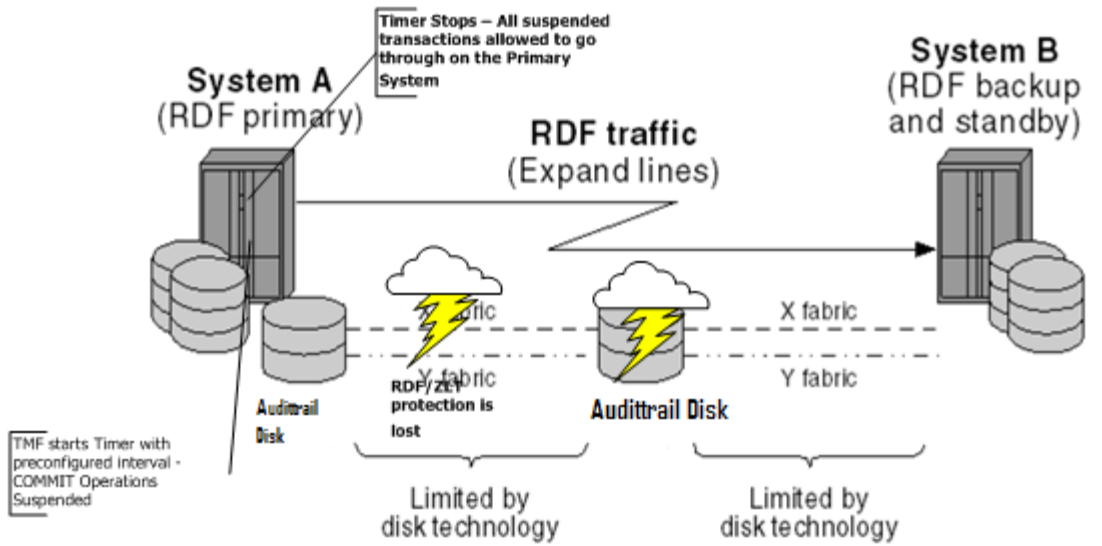
TMF Crash (Diagram continued after DP2 Communicates to TMF)

Figure 9: Timer Ends – TMF Crashes on RDF Primary System



Suspension of ZLT Protection (Diagram continued after DP2 Communicates to TMF)

Figure 10: Timer Ends – RDF/ZLT Protection is Lost



Technical Information

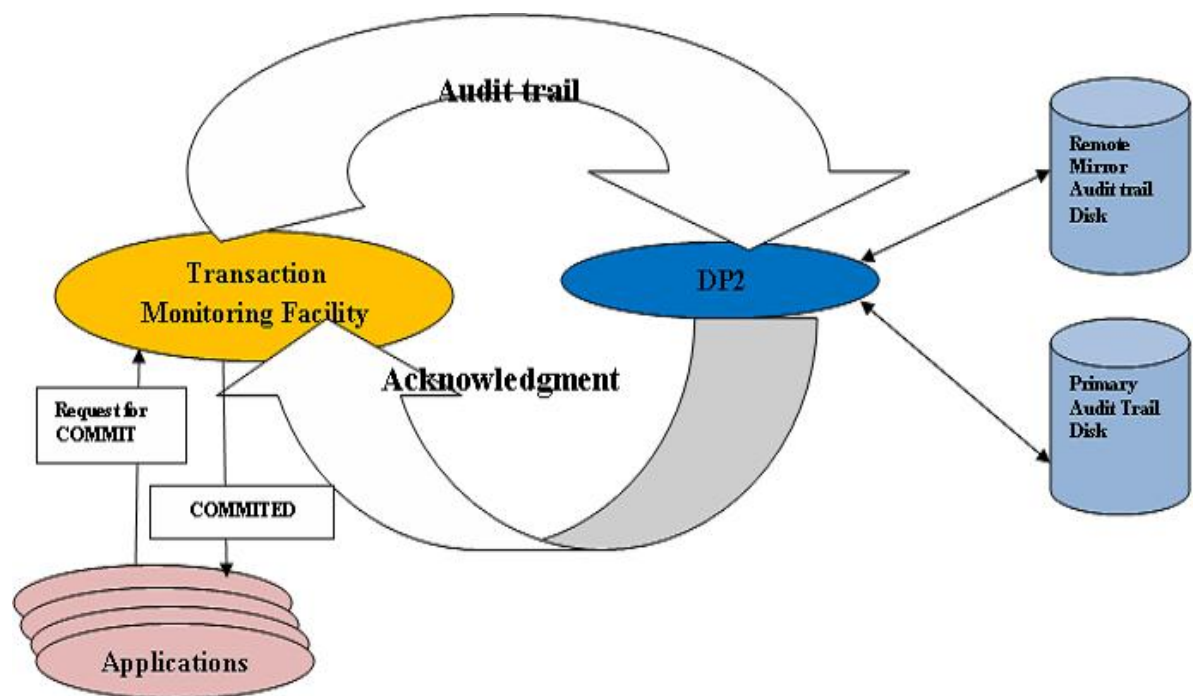
Subsystems Involved

The success of the COMMITHOLD operation depends on the interaction between:

- Transaction Management Facility (TMF)
- Disk Process (DP2)
- RDF/ZLT

How Applications Perceive COMMITHOLD ON Functionality

Figure 11: COMMITHOLD – Effects on application transaction



COMMITHOLD ON involves the following internal processing:

1. DP2 is informed that writing AT should first happen in the remote mirror AT disk and then in the primary AT disk.
2. Only after DP2 informs TMF that both writes have been successful, TMF reports success to the application calling the transaction COMMIT.

Some Permutations and Combinations

Based on the discussion so far, the interaction between TMF and DP2 subsystems with/without COMMITHOLD can be organized in four categories:

Successful write to both mirrors with COMMITHOLD OFF

DP2 can write to the remote or local (primary or backup) first and then writes to the other. When the writes have succeeded, DP2 tells TMF of the success, and TMF returns FEOK to End Transaction.

Successful write to only one mirror with COMMITHOLD OFF

The DP2 can write to the remote or local (primary or backup) first and then it writes to the other, as long as one write succeeds, then it tells TMF of the success, and TMF passes FEOK to End Transaction. Note: the damaged mirror is now declared down and receives no further attempts to write.

Successful write to both mirrors with COMMITHOLD ON

The DP2 always writes to the remote mirror first and then writes to the local; when both writes have succeeded, it tells TMF, and TMF returns FEOK to End Transaction.

Problem in writing to the local mirror with COMMITHOLD ON

DP2 successfully writes to the remote mirror first, ~~it succeeds,~~ but the write to the local gets an error. This is similar to the condition above: because the write to the remote succeeded, ~~the~~ DP2 tells TMF the write succeeded, and TMF returns FEOK to End Transaction.

Problem in writing to the REMOTE mirror with COMMITHOLD ON

DP2 writes to the remote mirror first, it gets an error. In this case it still proceeds with the write to the local mirror, but it tells TMF of the problem about the write to the remote.

- TMF does NOT return status to End Transaction. Note that ZLT protection is still active because, even though the write to the local succeeded, the caller of End Transaction has ~~still~~ not yet been told the outcome, thus the caller continues to wait for the outcome
- TMF starts the CHTIMER
- DP2 continues to process new audit, which is appended to the buffer for the remote mirror where the earlier audit of the failed write still resides. Once again, the remote is written ~~to~~ first, and if it still continues to have a problem, then DP2 writes the new audit to the local and continues to tell TMF of the problem
- More AT is generated, and there are several options
 - If the remote write still fails, DP2 tells TMF, and TMF finds that the CHTIMER timer has popped, then TMF will proceed according to its configuration:
 - If SUSPEND, it suspends COMMITHOLD processing, which means that normal non-COMMITHOLD operations proceed, as in “Successful write to only one mirror with COMMITHOLD OFF” above and FEOK is returned to all outstanding End Transaction calls. Note, callers of End Transaction get FEOK, but the user no longer has ZLT protection; they just have normal RDF protection where a failure of the primary system could lead to loss of some committed transactions.
 - If the next attempted write to the remote succeeds, then that write successfully flushes ALL the queued audit to the remote, DP2 tells TMF of the success, TMF turns off the timer, and TMF returns FEOK to all waiting End Transaction callers: there is still full ZLT protection.

Performance Discussion

There is NO difference in the writing technique between having COMMITHOLD ON or OFF.

For performance, the writes are parallel and no-wait, however it is essential to note that DP2 cannot initiate the two writes to the two mirrors at precisely the same time; one is started first and as soon as it is underway, the second is started. Then DP2 waits for both acknowledgments.

COMMITHOLD ON results in this difference in the internal processing behavior:

With COMMITHOLD ON, DP2 always initiates the first write to the remote mirror AT disk. For DP2, it is business as usual, except for ensuring that the first write goes to the remote mirror ~~first~~. The write must initiate to the remote first because, if the write contains a COMMIT, it must be safe on the backup system first, so that ~~then~~ it can be retrieved by RDF during a ZLT Takeover operation. With COMMITHOLD ON, DP2 waits to learn the outcome of both writes before notifying TMF.

With COMMITHOLD OFF, the initial write can go to either mirror first - this logic that has been used for decades now.

It is still possible that a disaster can crash the primary system in that split second after the writes have completed, but before TMF has replied with FEOK status to End Transaction. This has always been the case with regular RDF, as well as with Process Lockstep⁺ and ZLT.

⁺ When a process invokes the lockstep operation for a business transaction, the process must wait until all audit records associated with that business transaction are safely stored in image trails on the backup system before continuing.

Issues with Reducing Timeout Value for COMMITHOLDMODE

Reducing Timeout value can very easily lead to unnecessary suspension of COMMITHOLD. Note that, it in a suspend state, after a problem involving the remote mirror is fixed, the remote mirror needs to be revived from the local mirror, then when both mirrors are working in synch again - and only then - can the administrator enable COMMITHOLD again.

If the 5 second timer delay is uncomfortable because of the potentially high volume of transactions that could be backed up, remember that this is a necessary delay for the assurance of ZLT. One alternative is to fall back to the Process Lockstep protocol and eliminate delays.

Why We Cannot Support Only COMMITHOLD ON with No TIMER

Remember that RDF/ZLT ensures that no committed transactions will be lost and TMF has two main goals: protecting the consistency of a database and also ensuring no committed transaction is lost.

RDF/ZLT supports shipping AT to backup systems and ensuring that no committed transactions on the primary are lost on backup systems.

If no timer is present, then the very purpose of RDF/ZLT is defeated. When the primary goes down, there is always a without COMMITHOLD on, the primary and backup databases are not in sync. Instead of RDF/ZLT, Process Lockstep can be used.

Finally, At What Level Is COMMITHOLD Applied?

Commithold is applied at a volume level – and the timer does not stop if an AT rollover starts.

For more information

www.hp.com/go/nonstopcontinuity

Technology for better business outcomes

© Copyright 2009 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Linux is a U.S. registered trademark of Linus Torvalds. Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation. UNIX is a registered trademark of The Open Group.

4AA2-xxxxENW, February 2009

