SOFTEK
Simplifying Storage Management

Softek™ Replicator

Administrator's Guide for AIX, HP-UX, Linux, and Solaris

Version 2.1

ML–145086–001

www.softek.com

Administrator's Guide for AIX, HP-UX, Linux and Solaris
Version 2.1

Softek™ Replicator

SOFTEK
Simplifying Storage Management

ML–145086–001 12/03

REVISION NOTICE

This is the first release of this manual. A complete revision history is provided at the end of the manual.

ABSTRACT

The Softek Replicator 2.1 Administrator's Guide for AIX, HP-UX, Linux and Solaris (ML-145086) contains a conceptual overview of Softek Replicator, including the product architecture and operation. It tells you how to configure and start Softek Replicator on all systems. The product must be installed as directed in the Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux and Solaris (ML-145087) before the configuration can begin, This manual also covers all Softek Replicator administration tasks that are performed after the initial installation and configuration of the software, data recovery, monitoring tools, commands, and complex configurations.

FOR FURTHER INFORMATION

If you wish to obtain further information about the Softek Technology Corporation product discussed in this publication, contact your Softek marketing representative, or write to Softek Technology Corporation, Marketing Communications, Mail Stop 215, P.O. Box 3470, Sunnyvale, CA 94088-3470.

TECHNICAL SUPPORT

To obtain technical support on the Softek Technology Corporation product discussed in this publication, please call 1-800-66SOFTEK.

RESTRICTIONS ON USE

The information contained in this manual is the licensed property of Softek Technology Corporation. Use of the information contained herein is restricted pursuant to the terms and conditions of a license agreement.

About This Guide

This document contains information on how to configure and manage Softek Replicator. See the *Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux and Solaris (ML-145087)* for information on installing, upgrading, and removing the product. For information on product messages, see the *Softek Replicator 2.1 Messages Guide for AIX, HP-UX, Linux and Solaris (ML-145088)*.

Softek Replicator allows remote mirroring of volume based data over a LAN or WAN as a means to manage planned and unplanned events such as database migration, normal system maintenance, or disaster recovery. This manual describes how to configure and operate Softek Replicator on various UNIX platforms.

Audience

The information in this guide is intended for system administrators who are responsible for maintaining business-critical servers and their associated data. It assumes knowledge in the following areas:

- File systems and disks
- System startup files and directories
- Operating system processes
- Shell scripts
- Networks

Contents of this Manual

Chapter/Appendix	Description
Chapter 1: Introduction	What is Softek Replicator? Softek Replicator Overview, Softek Replicator Architecture, and Softek Replicator Operating Modes
Chapter 2: Configuration	Configuration for AIX, HP-UX, Linux, and Solaris; Common Configuration for AIX, HP-UX, Linux and Solaris; Configuring Softek Replicator with an RDBMS

Chapter/Appendix	Description
Chapter 3: Administering Softek Replicator	Creating Throttles, Changing Tunable Parameters, Adding/Modifying dtc Devices, Deleting Logical Groups, Relocating the pstore, Modifying the BAB, Resizing the Journal File Directory, Checkpointing Softek Replicator
Chapter 4: Recovering Data	Scenario 1: Secondary System/Network Failure Scenario 2: Primary System Failure Scenario 3: Failure to Transition Out of Tracking Mode Scenario 4: Failing Over to the Secondary System, Restoring the Primary System Scenario 5: Device Failure on the Primary or Secondary System
Chapter 5: Monitoring Tools	dtcperftool, dtcmonitortool, dtcinfo
Chapter 6: Commands	All Softek Replicator commands
Chapter 7: Complex Configurations	Symmetric Configuration, One-to-many, Chaining Loopback Configuration
Appendix A: Configuration Files	Configuration File Sample of a Primary Logical Group
Appendix B: Throttle Reference	Throttle Format, Measurement Keywords, Actions Action Directives, Tunable Action Parameters, Action Message Substitutions
Appendix C: Tunable Parameters	All Softek Replicator tunable parameters
Appendix D: Sample Script	Describes the sample scripts that can be used for changing the symbolic link to raw disks or volumes for Relational Database Management Systems.

Contacting Technical Support

At Softek, we work hard to provide products and service that anticipate and solve our customers' increasingly complicated application management challenges. In addition, we try to provide clear and easy-to-use online and printed documentation to enable you to work independently in managing application recovery issues.

If you have a technical issue that you can't answer with the provided resources, please contact Softek Technical Support by Telephone or the Web.

▶ **On the Web:**

1. Visit for :
   - Severity 2, 3, or 4 problems and tracking status of calls
   - Problem ownership and management throughout resolution
   - Asking questions

- Seeking product-specific information via FAQs, user libraries, patches, discussion forums, and to download documentation. License requests should come via your Account Executive.
- Submit enhancements requests
- Reviewing and updating your user contact and site location information.

2. Click the **Call Tracking Center** link.

NOTE You must have a User ID and Password to access the Softek Call Tracking and Problems Reporting database pages. To request access, go to the Call Tracking web page: .

▶ **For severity 1 problems and around-the-clock mission-critical support:**

Contact Softek Technology Corporation using the following phone number:

- Softek Global Support Center (World-Wide) 800-66SOFTEK (763835)
  - From North America, please dial 1-800-667-6383.
  - From Europe, please dial 00800-66 763835 (country code is not required).
    - From Austria, please dial 0800 200 236, then enter 05, and then 8006676383.
  - During weekends and holidays, you will be asked if you require assistance before the next business day. If so, Softek will respond within stated response time commitments.

NOTE Other phone numbers may be used within European countries where local language capability exists. Contact your local Softek Sales Representative. Major issues can be addressed to Softek management by a request through the Alert Centers.

Notices

The following notices are used throughout this manual.

CAUTION:
Alerts readers to a situation that could damage the software or interrupt operations.

NOTE Gives readers additional significant information about the subject to increase their knowledge or to guide their actions.

TIP Helps users apply the techniques and procedures described in the text to their specific needs.

Related Publications

The following publications contain related information.

Title	Part Number
Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux and Solaris	ML-145087
Softek Replicator 2.1 Messages Guide for AIX, HP-UX, Linux and Solaris	ML-145088
Softek Replicator 2.1 Administration and User Guide	ML-145090
Softek Replicator 2.1 Installation Guide for Windows	ML-145091

CONTENTS

Chapter 1: Introduction 1

What is Softek Replicator? 3
Product Highlights 3
Softek Replicator Implementations 4
Disaster Planning and Recovery 4
Data Migration 4
Backup 4
System Maintenance 4
Softek Replicator Overview 5
Softek Replicator Environments 6
Windows Servers only 6
Windows and UNIX Servers 6
UNIX Servers 7
Softek Replicator Architecture 7
Primary System 7
Softek Replicator Master Daemon: in.dtc 7
The PMD 7
Dtc Device Driver 8
The BAB 9
The Pstore 9
Logical Groups 9
Local Data Device 10
Dtc Device 10
Secondary System 11
The RMD 11
Mirror Device 11
Journal File Directory 12
Throttles 12
Tunable Parameters 12
Sync, Async, and Near Sync Modes 12
Softek Replicator Operating Modes 13
PASSTHRU Mode 13

NORMAL Mode 14
TRACKING Mode 14
REFRESH Mode 14
Full Refresh 15
Smart Refresh 15
Checksum Refresh 15
BACKFRESH Mode 15

Chapter 2: Configuration 17

Configuration for AIX 19
Configuring Softek Replicator 19
How to Define a Logical Group 20
How to Define dtc Devices 21
How to Define Throttles 22
Tunable Parameters 22
Primary and Secondary System Master Daemon Port Numbers 22
Distributing the Configuration Files 22
Configuring Softek Replicator with File Systems 24
Starting Softek Replicator 24
Mounting Journaled File Systems (JFS) on dtc Devices 24
How to Start Softek Replicator Mirroring 25
Using dtcoverride 25
Using launchrefresh 25
Using the Courier Transport Method 26
Configuration for HP-UX 27
Configuring Softek Replicator 27
How to Define a Logical Group 28
How to Define dtc Devices 29
How to Define Throttles 30
Tunable Parameters 30
Primary and Secondary System Master Daemon Port Numbers 30
Distributing the Configuration Files 30
Configuring Softek Replicator with File Systems 32
Starting Softek Replicator 32
How to Modify /etc/fstab 32
How to Start Softek Replicator Mirroring 33
Using dtcoverride 34
Using launchrefresh 34
Using the Courier Transport Method 34
Configuration for Linux 35
Configuring Softek Replicator 35
How to Define a Logical Group 36
How to Define dtc Devices 37
How to Define Throttles 38
Tunable Parameters 38
Primary and Secondary System Master Daemon Port Numbers 38

Distributing the Configuration Files 39
Configuring Softek Replicator with File Systems 40
Starting Softek Replicator 40
How to Modify /etc/fstab 40
How to Start Softek Replicator Mirroring 41
Using dtcoverride 41
Using launchrefresh 41
Using the Courier Transport Method 42
Configuration for Solaris 43
Configuring Softek Replicator 43
How to Define a Logical Group 44
How to Define dtc Devices 45
How to Define Throttles 46
Tunable Parameters 46
Primary and Secondary System Master Daemon Port Numbers 46
Distributing the Configuration Files 47
Configuring Softek Replicator with File Systems 48
Starting Softek Replicator 48
How to Modify /etc/vfstab 48
How to Start Softek Replicator Mirroring 49
Using dtcoverride 50
Using launchrefresh 50
Using the Courier Transport Method 50
Common Configuration for AIX, HP-UX, Linux and Solaris 51
How to Define Tunable Parameters 51
How to Change the Primary System Master Daemon Port Number 52
How to Change the Secondary System Master Daemon Port Number 53
Configuring Softek Replicator for
Relational Database Management Systems (RDBMS) 53
Moving a Non-Symbolic Linked Database to Softek Replicator 55

Chapter 3: Administering Softek Replicator 57

Creating Throttles 59
Editing Throttle Definitions 59
How to Use Throttle Builder 60
How to Build Expressions 60
How to Build Actions 61
Throttle Examples 62
Changing Tunable Parameters 63
Using dtcset 63
Using dtcconfigtool 64
Adding/Modifying Devices 65
Deleting Logical Groups 67
Relocating the pstore 68
Modifying the BAB 68
Using dtcconfigtool 68

Manually 68
Resizing the Journal File Directory 69
Checkpointing Softek Replicator 70
Examples of Checkpoints 70
Checkpoint Processing 70
Checkpoint Shell Scripts 71
Primary System Scripts 72
Secondary System Scripts 73
Checkpoint Scripts for File Systems 74
Primary Checkpoint Scripts 74
Secondary Checkpoint Scripts 75
Tips for Checkpoint Scripts 76
Considerations and Limitations 76
Mounting Mirror Devices on HP-UX and AIX 77
Rebooting the Primary System on HP-UX 77

Chapter 4: Recovering Data 79

Scenario 1: Secondary System/Network Failure 81
Scenario 2: Primary System Failure 82
Switching to the Secondary System with an RDBMS 82
Scenario 3: Failure to Transition Out of Tracking Mode 83
Scenario 4: Failing Over to the Secondary System, Restoring the Primary System . . 83
Scenario 5: Device Failure on the Primary or Secondary System 84

Chapter 5: Monitoring Tools 85

dtcperftool 87
Step I: Chart Setup 88
Step II: Measurements Shown in Chart 88
Step III: Chart Display 89
Modifying a Chart 89
Other Chart Functions 89
Performance Monitoring Files 89
Related Tunable Parameters 90
dtcmonitortool 90
Status Message Area 90
Error and Warning Messages 91
Logical Group / dtc Device Status area 91
Notification and Update Controls 92
dtcinfo 93

Chapter 6: Commands 95

dtcautomount 97
dtcbackfresh 98
dtccheckpoint 99
dtcconfigtool 100

dtcdebugcapture 101
dtchostinfo 101
dtcinfo 101
dtcinit 103
dtckillbackfresh 103
dtckillpmd 104
dtckillrefresh 104
dtckillrmd 105
dtclicinfo 105
dtcmonitortool 106
dtcoverride 106
dtcpmd 107
dtcperftool 107
dtcrefresh 108
dtcrmdreco 109
dtcset 110
dtcstart 111
dtcstop 112
dtcusersguide 112
killbackfresh 112
killdtcmaster 113
killpmds 113
killrefresh 114
killrmds 115
launchbackfresh 115
launchdtcmaster 116
launchpmds 116
launchrefresh 117

Chapter 7: Complex Configurations 119

Symmetric Configuration 121
Defining a Symmetric Configuration 122
On System B 122
On System A 122
Running B as Stand-Alone 123
Inverting the A-B Topology to B-A 124
Switching Back from A to B 124
One-to-many 125
Chaining 127
B to C Setup 128
A to B Setup 128
Loopback Configuration 129

Appendices

Appendix A: Configuration Files 131

Appendix B: Throttle Reference 135

Appendix C: Tunable Parameters 143

Appendix D: Sample Script 149

Glossary 153

Index 157

Revision History 161

Tables

Tunable Parameters 49
Sample Symmetric Definitions 118
Sample One-To-Many Definitions 121
Sample Chaining Definitions 124
Options 147

Figures

Components 5
The PMD/RMD Relationship 7
Position of the dtc Device Driver in the UNIX Kernel and its Relationship to Data Devices 7
Relationship Between Softek Replicator, Logical Groups, dtc Devices, Local Data Devices and Mirror Devices 8
Softek Replicator Operating Modes and State Transitions During Normal Operations 12
AIX Configuration- dtcconfigtool Introductory Message 17
AIX Configuration- Defining the BAB 17
AIX Configuration- Defining the Primary and Secondary Systems 18
AIX Configuration- Defining the dtc Devices 19
HP-UX Configuration- dtcconfigtool Introductory Message 25
HP-UX Configuration- Defining the BAB 25
HP-UX Configuration- Defining the Primary and Secondary Systems 26
HP-UX Configuration- Defining the dtc Devices 27
Linux Configuration- dtcconfigtool Introductory Message 33
Linux Configuration- Defining the BAB 33
Linux Configuration- Defining the Primary and Secondary Systems 34
Linux Configuration- Defining the dtc Devices 35
Solaris Configuration- dtcconfigtool Introductory Message 42
Solaris Configuration- Defining the BAB 42
Solaris Configuration- Defining the Primary and Secondary Systems 43
Solaris Configuration- Defining the dtc Devices 44
Configuring AIX, HP-UX, Linux and Solaris - Tunable Parameters Tab in dtcconfigtool 51
Configuring AIX. HP-UX, Linux and Solaris - TCP Settings 52
The Throttles Editor 57
Setting the Evaluation Time through the Throttle Builder 58
Throttle Expressions 58
Building Throttle Actions with the Throttle Builder 59
Throttle to Regulate Maximum Network Bandwidth during Peak Business Hours 60
Throttle to Restrict Network Usage during Peak Business Hours 61
Tunable Parameters Tab in dtcconfigtool 62
Script Sample of /etc/opt/SFTKdtc/cp_pre_on_p012.sh for Primary System 69
Script Sample of /etc/opt/SFTKdtc/cp_post_on_p012.sh for Primary System 69
Script Sample of /etc/opt/SFTKdtc/cp_post_on_s012.sh for Secondary System 70
Script Sample of /etc/opt/SFTKdtc/cp_pre_off_s012.sh for Secondary System 71
File System Checkpoint Script - /etc/opt/SFTKdtc/cp_post_on_s001.sh 72
File System Checkpoint Script - /etc/opt/SFTKdtc/cp_pre_off_s001.sh 72
Softek Replicator Performance Monitor Window 83
dtcmonitortool Window 86
Softek Replicator Group and Device Status Area of dtcmonitortool 87

dtcinfo Output Example 89
Sample Symmetric Configuration 115
Sample One-To-Many Configuration 119
Sample Chaining Configuration 121
Softek Replicator Configuration File: /etc/opt/SFTKdtc/p000.cfg 127
Example of Network Bandwidth Throttle 140

1

Introduction

What is Softek Replicator? 3

Softek Replicator Overview 5

Softek Replicator Environments 6

Softek Replicator Architecture 7

Softek Replicator Operating Modes 13

What is Softek Replicator?

Softek Replicator is a network-based disk mirroring product for AIX, HP-UX, Linux and Solaris that enables data exchange and synchronization in support of local applications, data sharing among remote sites, and disaster recovery.

In an outage scenario, Softek Replicator provides a recoverable copy of coherent application data—that is, a copy that can be accessed and used—on a secondary system.

This product mirrors disk-based data from devices on a primary system to disk devices on a secondary system, across any available network connection supporting TCP/IP. Data is duplicated in near-real-time to ensure integrity between the two systems in the event of hardware failure, natural disaster, or human intervention. Softek Replicator accomplishes this through time-sequenced transfers of data from the primary system to the secondary system over the network. If a failure occurs on the primary system, the secondary system can provide immediate access to contemporary business-critical data. Both the primary and secondary systems must have sufficient amounts of disk storage, and adequate network bandwidth to accommodate the flow of data. Softek Replicator operates with all disk subsystems supported by AIX 4.3.3, 5.1, 5.2; HP-UX 11 and 11i, Linux RedHat 9.0 (kernel 2.4.20-8), and Solaris 2.6, 7, 8, and 9.

Softek Replicator is compatible with UFS, NFS *Server*, VxFS, VxVm, as well as other common file systems, volume managers, DBMS, and disk utilities.

In environments, each disk should be partitioned and accessed as described in the *Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux and Solaris (ML-145087)*. Softek Replicator allows you to begin mirroring the existing database or file system data by simply incorporating the devices or volumes where this data is stored into the Softek Replicator configuration. You do not need to re-partition or reformat disks, to reinitialize file systems, or to import/export data.

In addition, Softek Replicator does not manipulate or corrupt user data in any way. Unlike other mirroring or replication programs, Softek Replicator performs continuous transfers of updated data to the mirror site, minimizing data loss and recovery time should a failure occur. Softek Replicator ensures that users have uninterrupted access to a usable data set.

Product Highlights

The Softek Replicator product:

- Requires no additional hardware, providing there is sufficient physical memory and disk space for your mirrored data.
- Provides continuous data mirroring over LAN or WAN, or locally through a loopback connection.
- Maintains data coherence on primary and secondary systems.
- Significantly reduces data loss during outages.
- Recovers automatically from network outages.
- Supports planned events such as database migration, “hot” backups, normal system maintenance such as upgrading operating systems.

Softek Replicator Implementations

This product is uniquely applicable as a data management solution for both planned and unplanned events. This section gives a brief description of a few pertinent applications.

Disaster Planning and Recovery

Softek Replicator's ease of integration into existing environments and the flexibility of configuration make it the right choice for disaster planning and recovery. Softek Replicator ensures user access to mission-critical data even when the primary system goes down by providing a continuously updated copy of data on a secondary system. Softek Replicator's approach to continuous mirroring of data across the network drastically reduces data loss in the event of a disaster, and greatly simplifies recovery operations.

Data Migration

When upgrading or moving your data center, it is common to install and prepare a new server while the existing server continues to provide service. Even with parallel systems, the downtime can be lengthy when copying, transferring and restoring data from an old server to a new one. This downtime is significantly reduced by using Softek Replicator to create a mirror and continuously send changes to the new server until you are ready for a switch-over.

Backup

Softek Replicator complements tape backups. By using a secondary data set on a remote server, business-critical resources at the primary location are kept available while a tape backup is performed on the secondary site. Checkpointing is a mechanism for implementing a "hot backup" environment. For more information, refer to *Checkpointing Softek Replicator* on page 70.

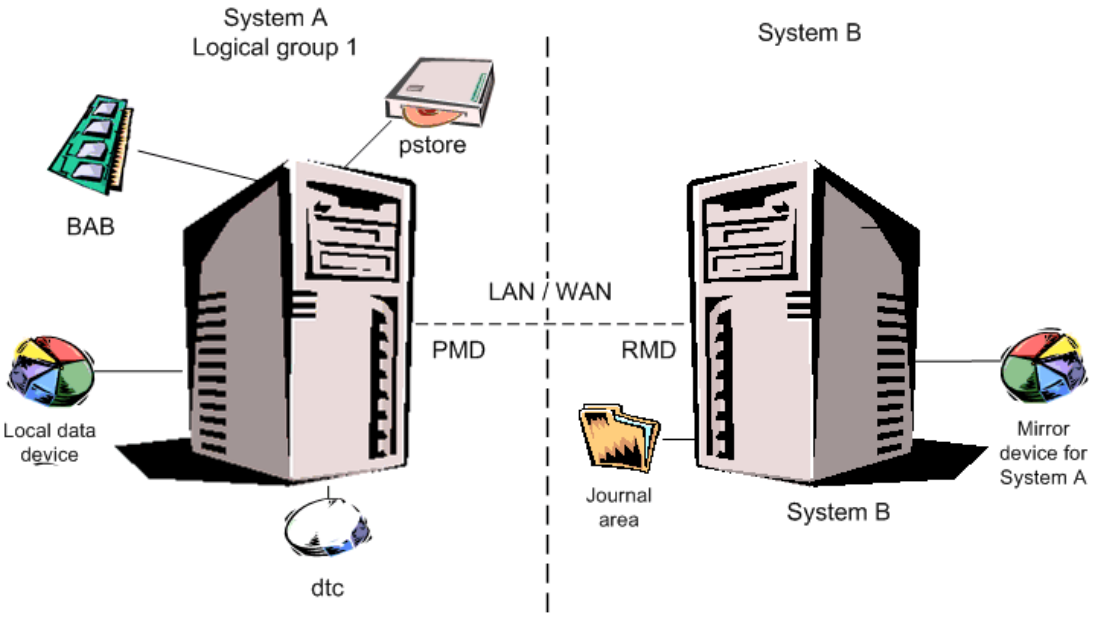
System Maintenance

Softek Replicator can be used to relocate data and applications to a secondary system while the primary system receives normal system maintenance, such as upgrading the operating system or application software, adding additional storage and memory, or even replacing the primary system with a newer model.

Softek Replicator Overview

Softek Replicator is installed on all systems in the configuration. Softek Replicator installs a special *dtc device driver* just architecturally above the actual disk device drivers or volume management device drivers, but below file systems or applications. As a result, any disk-based file system supported by the operating system is compatible with Softek Replicator, as are applications that work directly with raw disk devices.

Components



As the figure: *Components* shows, Softek Replicator uses a kernel memory-based buffer cache - called the *BAB* (Big Asynchronous Buffer) - on the primary system to track all updates made to the *local data device.* As data is written to the local data device, the same information is stored in the *BAB.*

Entries are then read from the BAB by the *Primary Mirror Daemon* (*PMD)* and transferred across the network connection to the *Remote Mirror Daemon* (*RMD*) on the secondary system. The PMD is a background daemon program running on the primary system; Softek Replicator creates one PMD for each *logical group*. The RMD writes data received from the primary system to the *mirror devices.* You configure a mirror device on the secondary system for each local data device on the primary system.

The *dtc device*, which you define during configuration, provides the mapping to and management of a specific local data device, as well as the corresponding mirror device(s). You can define a collection of dtc devices as a coherent unit, called a *logical group*. Each logical group operates with its own independent PMD/RMD daemon pair.

During normal operation, updates are written directly to the mirror device on the secondary system. Data is not transformed, manipulated or corrupted by Softek Replicator in any way.

However, you configure a *journal file directory* on the secondary system (optional: one for each logical group) to handle situations when the data being transferred is not chronologically ordered, such as during a smart refresh operation. (During a smart refresh, only blocks of data that have changed are mirrored from primary to secondary devices.) If this data were to be applied to the mirror device directly, it would make data on the mirror incoherent.

The journal file directory is also actively receiving updates during *checkpoint* of the data set on the mirror device. Checkpoint allows you to take the secondary mirror off-line from Softek Replicator, such as maintenance, and make it available for read-only access by other applications.

In the standard configuration, Softek Replicator accumulates disk updates in the BAB independently of transmissions made to the secondary system. This dissociation is termed Async mode. The advantage of this configuration is that applications have near normal I/O performance, while allowing the daemon processes to exploit network bandwidth in an optimal way.

However, if you need an exact mirror of the data on the primary system available at all times, you can also configure Softek Replicator to operate in Sync and Near Sync modes. For more information, see *Sync, Async, and Near Sync Modes* on page 12.

Softek Replicator Environments

Softek Replicator supports the replication and migration of data across multiple operating systems (Windows and UNIX). As such, a Softek Replicator environment can use any of the following:

- Windows Servers only
- Windows and UNIX Servers
- UNIX Servers only

Windows Servers only

If your Softek Replicator environment is composed entirely of Windows servers, then all servers in your environment may be viewed and manipulated from a single **Common Console**. For more information, please refer to the *Softek Replicator 2.1 Administration and User Guide (ML-145090)*.

Windows and UNIX Servers

Softek Replicator supports the replication and migration of data across multiple operating systems, and through the Windows-based **Common Console**, data on each of these operating systems can be viewed and manipulated. Thus, information on every Replication Server can be viewed from a single source, regardless of the operating system. This reduces the need for redundant monitoring and administration over separate systems and machines. To do this, you will need to install the Softek Replicator 2.1 **Data Collector and Database** and **Common Console** components in your replication environment, as well as a Softek Replicator Agent, on each UNIX Replication Server machine (both the primary and secondary systems) that you wish to manipulate with the **Common Console**.

For more information on installing the Softek Replicator 2.1 **Data Collector and Database** and **Common Console** components, please refer to the *Softek Replicator 2.1 Installation Guide for Windows (ML-145091)*.

For more information on installing Softek Replicator Agents, please refer to the *Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux, and Solaris (ML-145087)*.

For more information on the Softek Replicator **Common Console**, please refer to the *Softek Replicator 2.1 Administration and User Guide (ML-145090)*.

UNIX Servers

If your replication environment includes only UNIX servers, you can use either the Windows-based **Common Console** to set up and manage your replication environment, or you can install Softek Replicator for UNIX on each replication server. Further still, by installing Softek Replicator Agents on your UNIX Replication Servers (both primary and secondary systems), you can set up and manage your replication environment using both the **Common Console** and Softek Replicator for UNIX.

For more information on installing Softek Replicator Agents, please refer to the *Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux, and Solaris (ML-145087)*.

For more information on the Softek Replicator **Common Console**, please refer to the *Softek Replicator 2.1 Administration and User Guide (ML-145090)*.

Softek Replicator Architecture

Softek Replicator consists of several components on the primary and secondary systems, which are described in detail in this section.

Primary System

The *primary system* provides primary application and data storage services to users. During initial installation and normal operation, this system runs applications and provides access to local data. All systems in the Softek Replicator environment must be running compatible versions of Softek Replicator.

The *dtc device driver,* the *local data devices*, *BAB* and *pstore* are located on the primary system. A master daemon process, called *in.dtc*, runs on all primary as well as secondary machines in the Softek Replicator enterprise. This master process launches and manages other processes on the primary system, including the PMD.

Softek Replicator Master Daemon: in.dtc

The *in.dtc* is a user-mode background program running on both the primary and secondary systems that is launched during the install process. *in.dtc* establishes and manages PMD and RMD processes for each defined logical group. This daemon also launches the process that manages and evaluates *throttles* for each *logical group* of devices. For information, see *Throttles* on page 12.

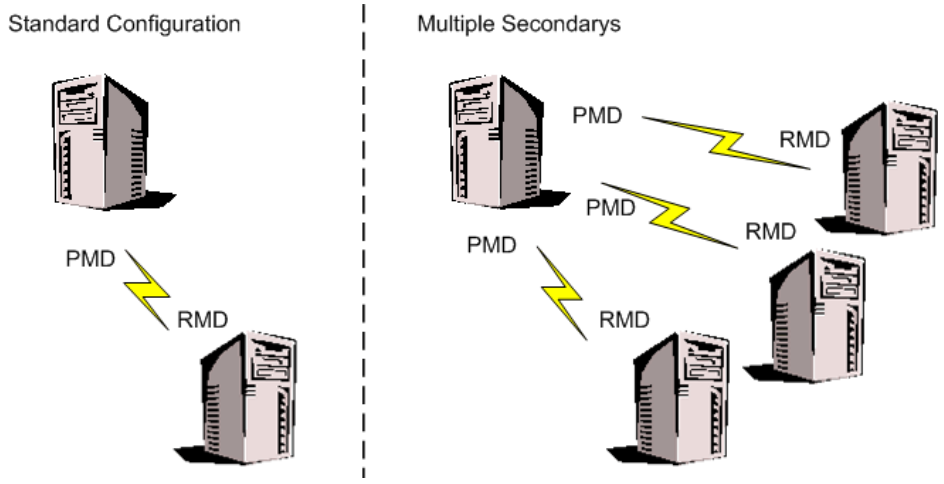
The PMD

The *Primary Mirror Daemon*(PMD) is a background process running on the primary system. One PMD for each logical group is started automatically by the master daemon as part of the system bootscript or using the `launchpmds` command.

The PMD drains entries out of the BAB and sends them to the corresponding RMD on the secondary system (for a description of the RMD, see *The RMD* on page 11.)

There is an independent PMD process for each logical group defined on the primary system, so each logical group can have a connection to a unique secondary system.

The PMD/RMD Relationship



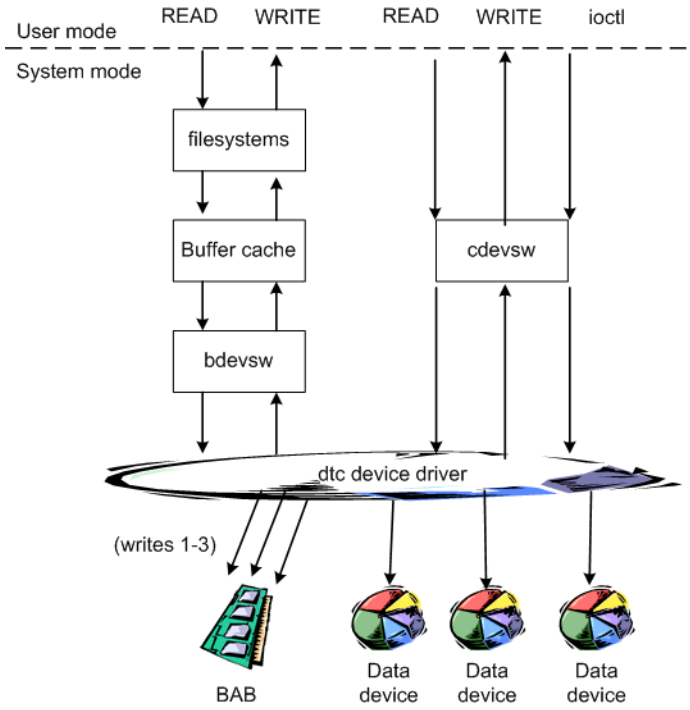
Dtc Device Driver

As the following figure shows, the *dtc device driver* is installed just above the actual disk device drivers or volume management device drivers, but below file systems or applications. It supports block and special character devices, which provide services to the kernel and user.

Block device drivers are usually limited to transferring blocks of a fixed size and use the buffer cache as an interface between the driver and the user application or file system.

The special character device allows the dtc device driver to be addressed in units smaller or lager than the device block. These transfers are performed independently of the file system or buffer cache, and allow the kernel to transfer data directly to or from the underlying device.

Position of the dtc Device Driver in the UNIX Kernel and its Relationship to Data Devices



The BAB

Softek Replicator uses a large kernel buffer called the Big Asynchronous Buffer (*BAB*) to journal updates as they are written to the local data devices when the system is in NORMAL mode. This buffer resides in physical kernel memory, owned by the operating system, and must be allocated to Softek Replicator during initial configuration.

Writes to a dtc device are simultaneously logged in the BAB and written to the local data device. Chronological coherence is maintained on a *logical group* basis, therefore each logical group assumes a portion of the BAB and updates to the devices in that group are journaled sequentially within the area allocated. The entire memory designated for the BAB is allocated dynamically (on an as needed basis) among groups. During configuration, you specify the memory allocated for the BAB.

The Pstore

A persistent store (*pstore*) is a disk volume that you specify on the primary system which stores state information, tunable parameter definitions, and tracking data. During configuration, you can define one pstore for all logical groups or for each one. The default setting is one pstore device, but multiple pstores are recommended for a high availability environment.

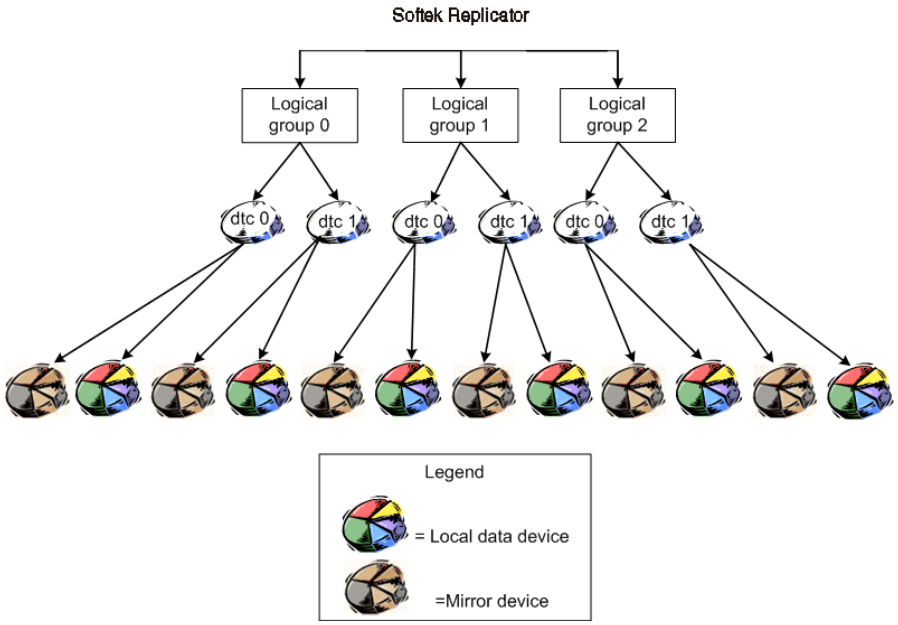
Changes are saved in the pstore with a controlled shutdown. The pstore is also used to perform a smart refresh operation if the primary system fails. This method of tracking updates ensures that no data is lost - particularly data in the BAB waiting to be mirrored.

The size of the pstore is directly related to the maximum number of devices, either in total or in the logical group, if separate pstores are defined. You should allow 140 KB per device for the pstore. Pstores sized larger than 100 MB are unnecessary.

Logical Groups

As the following figure shows, you can tell Softek Replicator to treat a collection of dtc devices as a coherent unit, called a *logical group*. Each logical group operates with its own independent PMD/RMD daemon pair. Logical groups allow time-ordered writing coherence between member dtc devices, and complete operational and state independence between logical groups.

Relationship Between Softek Replicator, Logical Groups, dtc Devices, Local Data Devices and Mirror Devices



You may want to define multiple logical groups for the following reasons:

- Some applications, especially databases, can work with several disk devices at the same time. It is essential that chronological coherence be maintained between all dtc devices so that they are in the same state. You can maintain chronological coherence of I/O transfers by organizing the dtc devices into a logical group.
- Individual logical groups can be targeted independently to secondary systems, thus creating a one-to-many configuration where a portion of the original data set resides on each secondary system. For more information on this and other configuration options, refer to *Complex Configurations* on page 119.
- Logical groups can utilize independent network connections to the secondary system, thus creating an aggregated throughput greater than that of any single network connection.
- The failure of one logical group does not affect the operations of the others.

Local Data Device

A *local data device* is a disk volume or a managed volume associated with the primary system. A local data device is specified as a special character device and acts as the interface to the block mode device. Data stored on a local device is duplicated onto the secondary system(s), so the local device must have a correlating mirror device of equal or greater size on the secondary system.

You specify the local devices to be included in your company's Softek Replicator profile during configuration.

Dtc Device

The *dtc device* is the means by which applications or file systems interact, access, or store information within Softek Replicator. A dtc device provides the mapping to and management of a specific local data device and the corresponding mirror device(s).

During configuration, you give each dtc device instance a unique name within a logical group—for example, dtc0, dtc12, dtc93. Device names usually begin with 0 and are incremented by 1.

Dtc devices appear as volumes to the kernel, so a dtc device accepts and handles any request that can be made to a normal volume or fixed-size volume, such as create and mount a file system, or allocate DBMS tablespace.

Dtc devices are not shared by the primary and secondary systems. Rather, data is mirrored across the network from the local data devices to the mirror devices. If you want the secondary system to assume all activities if the primary system fails, then install application software on both systems. In a standard Softek Replicator configuration, applications on the secondary system must not be executed until the system is prepared to act as the application server. The exception is when *checkpoint* is active. For more details on preparing for a changeover to the secondary system, see *Recovering Data* on page 79.

Secondary System

The *secondary system* is the system on which a copy of the primary system's local data is stored. The secondary system must be running the same operating system as the primary one.

During normal operation, this system provides a mirror of the data. There is no difference between the Softek Replicator software installed on primary system and the one installed on secondary system.

The functionality of each system is defined in the configuration files, which you define on the primary system and copy to the secondary one. This allows you to switch the roles of systems in Softek Replicator more easily. All applications used on the primary system should also be installed on the secondary system to allow switch operations over to the secondary system if needed. Note, however, that Softek Replicator does not perform automatic changeover.

Mirror devices and journal file directories are set up on the secondary system. The master process (*in.dtc*) running on the secondary system launches an RMD for each logical group.

There can be more than one secondary system in the configuration, each representing the entire original data set or a portion of the original data set. For more information on advance configurations, refer to *Complex Configurations* on page 119.

The RMD

The Remote Mirror Daemon (*RMD*) is a background process running on the secondary system launched by *in.dtc*. The RMD writes data received over the network from the primary system to the mirror devices. An RMD is automatically brought down when the corresponding PMD is killed.

Mirror Device

A *mirror device* is specified as a special character device (but pertains to both the special character and corresponding block mode devices) on the secondary system. You must have a mirror device on the secondary system for each local data device on the primary system. The mirror device must be the same size or larger than the corresponding local data device.

During normal operation, the mirror devices contain a coherent—that is, usable—copy of the data stored on the primary system. You can checkpoint the data on these devices, and applications other than Softek Replicator can open mirror devices in read-only mode. For more information on checkpoint, see *Checkpointing Softek Replicator* on page 70.

Journal File Directory

The *journal file directory* is used during a smart refresh operation, to store updates which, if applied to the mirror device, would place it in an incoherent state. The journal file directory is also used to store updates while the checkpoint is active on the mirror devices. These changes are eventually applied to the mirror device to maintain coherence. These entries are saved to the mirror device only after the PMD indicates that all updates have been transferred.

You can set the journal file directories to any writable directory during configuration.

A new journal file is created each time there is a state change that causes the data being transferred to go from coherent to incoherent (for example, during a *backfresh*). Softek Replicator uses the following naming convention for these files:

j###.###.c (for coherent transfer of data)

j###.###.i (for incoherent transfer of data)

The "j" indicates that this is a journal file, the first three numbers represent the logical group to which the journal file belongs, and the second set of numbers are sequence numbers of journal files for each logical group. For example:

j001.002.c (second coherent data journal file for logical group 1)

Throttles

Throttles help you to automate administration of Softek Replicator. During configuration, you can define a throttle to alert you to system trends or problems. Throttles help keep a machine operating within a defined range.

Throttle definitions are defined for each logical group. Softek Replicator evaluates throttles periodically, based on a tunable parameter, and performs the specified actions.

You can define an unlimited number of throttles for each logical group. Each throttle can have up to 16 ACTIONS executed when the throttle evaluates TRUE. In addition, there can be up to 16 clauses, or linked tests, in the throttle definition.

For details on throttles, refer to *Throttle Format* on page 137.

Tunable Parameters

Tunable parameters allow you to control aspects of Softek Replicator's function, such as the maximum size of a data packet sent from the primary system to the secondary system. Most of these parameters are saved in the pstore, and are frequently evaluated by the PMD. Some tunable parameters are only processed by the throttle, since they are part of a throttle definition. For more information on specifying tunable parameters, see *How to Define Tunable Parameters* on page 51.

Sync, Async, and Near Sync Modes

By default, Softek Replicator operates in Asynchronous Mode (*Async mode*), accumulating disk updates in the BAB that occur independently of the transmission of these entries to the secondary system.

You can also configure Softek Replicator to operate in Synchronous mode (Sync mode) or Near Synchronous Mode (Near Sync mode) with the SYNCHMODE tunable parameter. *Sync mode* does not return control to an application until after a disk update has been committed to both the primary system's local data device and the secondary system's mirror device. However, if Sync mode timeout occurs (the PMD did not issue a send completion response within the

time specified in SYNCMODETIMEOUT), the control will be returned to the application and the Logical Group in which the timeout occurred will be switched to TRACKING mode. In Sync mode, the secondary mirror disks are an exact copy of the data disk devices at all times, except when a SyncMode timeout occurs. In the case of SyncMode timeout, the secondary mirror disks will have an exact copy when the Logical Group Status reverts to NORMAL mode after the Smart Refresh.

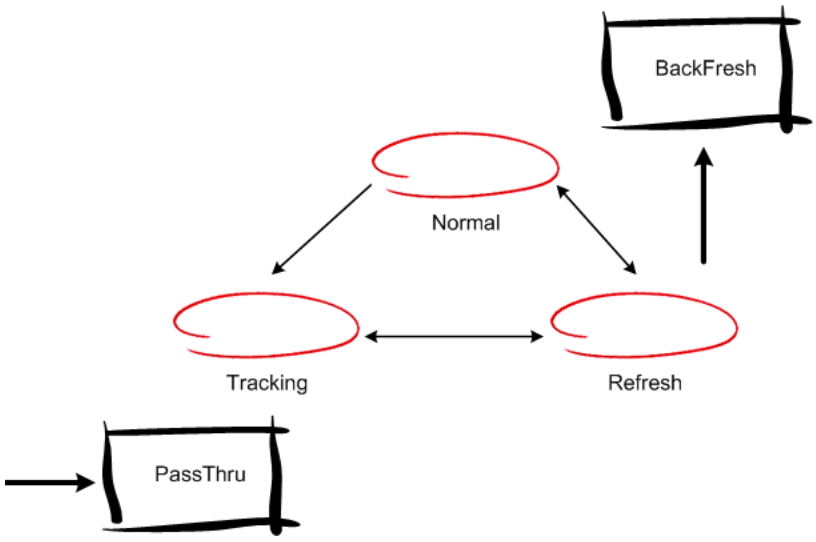
UFS files systems mounted on top of a Softek Replicator device in Sync mode continue to perform updates of I/Os asynchronously. VxFS file systems honor Softek Replicator's Sync mode if installed with `-o mincache=dsync`.

Near Sync mode is a middle ground between Async and Sync mode. In Near Sync mode, disk updates accumulate in the BAB asynchronously until they reach the limit allowed, at which time I/O operations are blocked by the application to allow updates to be written to the secondary system, until the entries in the BAB fall below this tunable limit. Near Sync mode reduces the Sync mode performance, yet ensures that the data on the secondary system's mirror device(s) is no more than a fixed number of disk updates behind the primary system.

Softek Replicator Operating Modes

The figure: *Softek Replicator Operating Modes and State Transitions During Normal Operations* shows Softek Replicator operating modes during normal activity. This section describes the different operating modes and Softek Replicator transitions in and out of each one. You can see what operating mode the product is in, using the Softek Replicator monitoring tool `(dtcmonitortool)` or the `dtcinfo` command.

Softek Replicator Operating Modes and State Transitions During Normal Operations



PASSTHRU Mode

Softek Replicator is in PASSTHRU mode by default when installed. In PASSTHRU mode, read/write requests are fulfilled by Softek Replicator devices, but the dtc device driver writes only to the local data device. Updates are not transmitted to the BAB and no data is mirrored to the secondary system.

After configuring the product, perform a full refresh to create the initial mirror and move Softek Replicator into NORMAL mode. Refreshing the set of data is the standard method to move a logical group from PASSTHRU mode to NORMAL mode.

NORMAL Mode

NORMAL mode indicates that Softek Replicator is installed; dtc devices handle read/write requests; updates are applied to local data devices and written to the BAB for transfer to the mirror devices on the secondary system, through the PMD/RMD daemons. In this mode, Softek Replicator performs continuous mirroring to have a coherent, recoverable copy of data on the secondary system.

TRACKING Mode

TRACKING mode reduces the need for a full refresh in the case of network outage or loss of communication with the secondary system's mirror devices. In TRACKING mode, Softek Replicator directs reads and writes to the dtc devices. Modifications to local data devices are also tracked. The pstore is updated on every appropriate I/O operation. The pstore consistently tracks all changes while Softek Replicator is in NORMAL, REFRESH, and TRACKING modes.

Entries are not written to the BAB, although the BAB may contain entries that have not been migrated from an earlier NORMAL mode environment. Updates to the dtc device received while in TRACKING mode are not mirrored to the secondary system, but are written to the local data devices. When Softek Replicator leaves this mode, the PMD uses the changes in the pstore to perform a smart refresh. A smart refresh is the standard method of transition from TRACKING mode to NORMAL mode.

Leaving the TRACKING mode without performing a refresh operation—either smart or full—causes the systems to be out of synchronization.

REFRESH Mode

You use the REFRESH mode to create an initial mirror or to synchronize the secondary system with the primary one. The REFRESH mode is a background operation, which means that it does not interfere with data mirroring. You place Softek Replicator in REFRESH mode with the `launchrefresh` command.

A REFRESH mode transition also occurs automatically if the BAB becomes full while in NORMAL mode. In this case, the system automatically goes into TRACKING mode, then to REFRESH mode. When the refresh operation is complete, the system goes back to NORMAL mode.

You can specify a refresh of single or multiple logical groups, depending on the option you specify on the refresh command. If you specify no options, a smart refresh is performed on all logical groups.

A refresh is stopped automatically when the refresh process is complete and Softek Replicator transitions to NORMAL mode. You can also explicitly kill a refresh operation with the `killrefresh` command. However, killing a refresh operation before it was complete causes a loss of synchronization of the primary and secondary systems.

Full Refresh

A *full refresh* mirrors every block on the designated local data device(s) to the secondary system. You use this method to create an initial mirror. Data is not transmitted to the secondary system during a full refresh operation, but is applied directly to the mirror devices. The data is coherent only when the full refresh is complete.

Smart Refresh

A *smart refresh* (the default refresh method) mirrors only those blocks on the local data device that have changed. Softek Replicator uses the information stored in the pstore to track changes made to the local data device(s). When the BAB becomes full, the system automatically transfers to TRACKING mode and then to REFRESH mode. When the smart refresh completes, the system goes back to NORMAL mode.

During a smart refresh, the data is written to journal files on the secondary machine. These entries are saved on the mirror device after all updates have been transferred. Data on the mirror device is incoherent while the journal entries are being applied. The data is coherent only when all entries have been saved to the mirror device.

Checksum Refresh

A *checksum refresh* compares all blocks on the local data device with the mirror device, using a checksum method to identify deltas. Only blocks that have been modified (that is, those for which the checksum varies) are sent to the secondary system. A checksum refresh writes mirror data to the journal file system, not directly to the mirror device.

BACKFRESH Mode

BACKFRESH mode synchronizes the primary system back from the secondary system. Backfresh is useful when you are performing maintenance on the primary system. A backfresh operation moves all blocks of data on the secondary mirror devices that differ from those on the local data devices to their corresponding locations on the primary system. Softek Replicator compares the blocks on the primary system with those on the secondary system to detect changes, using a checksum method.

While the backfresh operation is running, the applications should not access primary data devices and secondary mirror devices because the backfresh operation requires exclusive access to them. For more information on re-synchronizing your data using backfresh, see *Recovering Data* on page 79.

2

Configuration

Configuration for AIX 19

Configuration for HP-UX 27

Configuration for Linux 35

Configuration for Solaris 43

Common Configuration for AIX, HP-UX, Linux and Solaris 51

Configuring Softek Replicator for Relational Database Management Systems (RDBMS) 53

This chapter tells you how to configure and start Softek Replicator on all systems in the environment. The product must be installed as directed in the *Softek Replicator 2.1 Installation Guide for AIX, HP-UX, Linux and Solaris (ML-145087)* before configuration can begin.

Configuration for AIX

Configuring Softek Replicator

Use `dtcconfigtool` to create Softek Replicator configuration files. Run it on the primary system only.

▶ To start the configuration tool:

- Enter the following at the system prompt:
  `/usr/dtc/bin/dtcconfigtool`

The first time `dtcconfigtool` is run, it displays an informational message shown in the following figure that explains the BAB needs to be defined.

AIX Configuration - dtcconfigtool Introductory Message

Select **Ok** to dismiss this message and then take the following steps:

1. Allocate memory for the BAB by using the up and down arrows, or simply position the cursor in the box and enter a new value.

NOTE **Sizing the BAB on AIX (32bit kernel)**

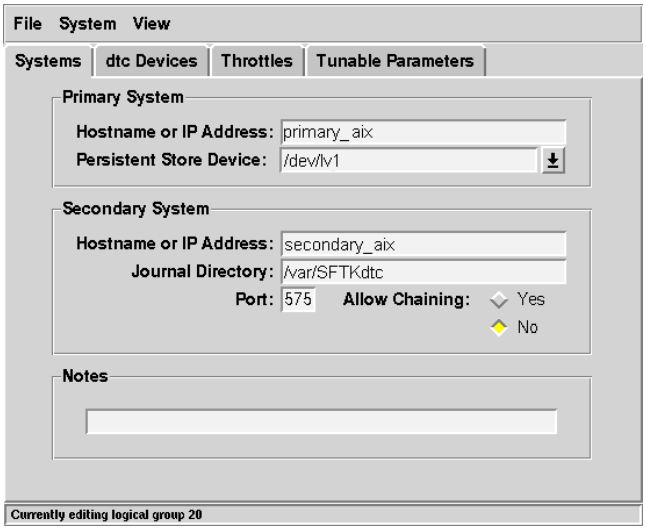
On systems running the AIX 32bit kernel, the maximum kernel memory size is 512MB, out of which a maximum of 256MB may be allocated to the BAB. If you allocate more than 256MB of kernel memory to the BAB, and depending on the processes and applications that are running on the system at a given time, you may encounter insufficient memory errors when you attempt to start a new process. If you encounter issues with kernel memory, Softek recommends that you decrease the BAB size.

AIX Configuration - Defining the BAB

2. Select **Ok**.

   `dtcconfigtool` opens the following dialog box.

AIX Configuration - Defining the Primary and Secondary Systems



How to Define a Logical Group

`dtcconfigtool` creates configuration files (.cfg) for each logical group. When you initially start the tool, you automatically edit Logical Group 0.

▶ To create a Logical Group other than 0:

1. From the **File** menu, select **New Logical Group**.
2. Use the arrows to select the Logical Group number for editing.
3. Under **Primary System**, the primary system **Hostname or IP Address** is filled in automatically with the name of the machine you are running on. Accept this, or enter a valid system name or IP address in this field.
4. Define the **Persistent Store Device** (**pstore**) for the group. You can define a unique pstore for each logical group.
5. Under **Secondary System**, specify either a resolvable system name or IP address for the secondary or mirror system.
6. To configure a local loopback configuration (that is, where both the primary and secondary are on the same system) by specifying `localhost` or `127.0.0.1` for the primary and secondary systems.

7. Keep in mind that each logical group has an independent connection, so you can define a different secondary system for each group.

8. If you specify a port number other than the default of 575 on the secondary system, you must enter that number in the **Port** field. This is the port you are connecting to on the secondary system. To change this port number, see *How to Change the Secondary System Master Daemon Port Number* on page 53.

   You can also change the port that the Softek Replicator master daemon listens on as described in *How to Change the Primary System Master Daemon Port Number* on page 52.

NOTE You can define logical groups that connect to different secondary systems with different port numbers. Use this field to specify those secondary port numbers.

9. In **Journal Directory**, specify any writable directory on the secondary system for journal files.

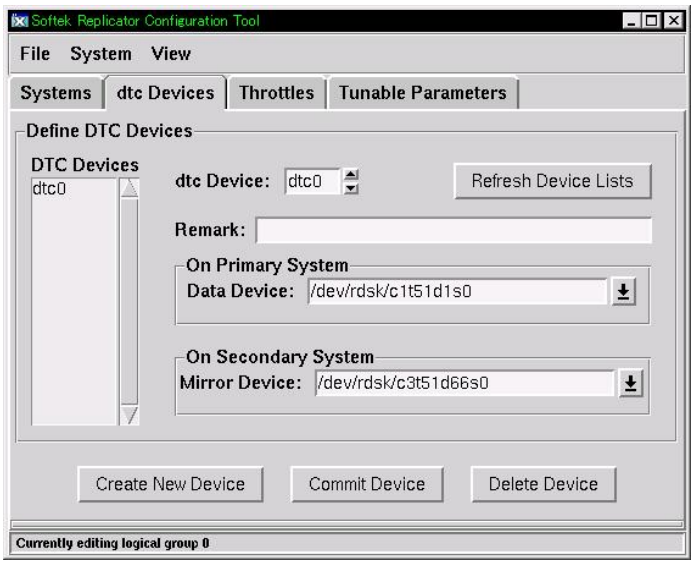
   After you complete the configuration, make sure you specify this directory in the system so that the configuration file is installed automatically at boot up. See *Mounting Journaled File Systems (JFS) on dtc Devices* on page 24.

10. If needed, change the **Allow Chaining** option (**No** is the default) for each secondary system defined. For more information on chaining, refer to *Chaining* on page 127.

How to Define dtc Devices

1. Select the **dtc Devices** tab, shown in the following figure.

AIX Configuration - Defining the dtc Devices



2. Choose a data device from the drop-down menu. You can see volumes and all started dtc devices in the data device and mirror device drop-down menus.

NOTE Dtc device names are automatically incremented by one when a new device is created, but you can change these names. Dtc device names must be unique only within a specific logical group. Multiple logical groups can contain identical dtc device names.

3. Choose a mirror device from the drop-down menu.

NOTE Each mirror device must be at least as large as its corresponding local data device.

4. Once you have defined local data and mirror devices, select **Commit Device**.
5. To add more dtc devices to the current logical group, select **Create New Device**.
6. Repeat these steps to create all logical groups and dtc devices, then exit `dtcconfigtool`. As you exit each logical group, you are prompted to save the group definition.

How to Define Throttles

Throttles are an optional part of the Softek Replicator configuration. For instructions on creating throttles, see *Creating Throttles* on page 59 and *Throttle Reference* on page 135.

Tunable Parameters

Tunable parameters are an optional part of the configuration. To define these parameters now, see *How to Define Tunable Parameters* on page 51.

Primary and Secondary System Master Daemon Port Numbers

The default port number that the master daemon (in.dtc) uses to listen is 575. You can change this port on the primary system. Remember that the port number on the secondary system needs to match the one on the primary system. If you wish to change the port number now, see *How to Change the Primary System Master Daemon Port Number* on page 52, and *How to Change the Secondary System Master Daemon Port Number* on page 53.

Distributing the Configuration Files

The configuration files must reside on both the primary and secondary systems, since each logical group contains definitions for dtc devices, including mirror devices. Softek Replicator uses the following naming convention for these files:

`[p|s]###.cfg` — where ### indicates the logical group number. The `p` and `s` indicate the primary and secondary systems.

You must copy the p###.cfg files onto all relevant secondary systems and rename them to s###.cfg.

1. Copy these files (using **ftp** or **rcp**) from the /etc/dtc/lib directory on primary to the same directory on the secondary system,
2. Rename the file on the secondary system, replacing the `p` with an `s`.

For example:

```
/etc/dtc/lib/p000.cfg (primary)
```

is copied and renamed to

```
/etc/dtc/lib/s000.cfg (secondary)
```

Since all logical group configuration files are stored in the same directory, the renaming allows each system in Softek Replicator to operate as both primary and secondary system without conflict.

For more information on advanced configuration options, refer to *Complex Configurations* on page 119.

NOTE If you change any primary configuration file, you must copy the updated file to the secondary system. Copy the file, then stop and restart the logical group. The `dtcstart` command processes the.cfg files, at which point changes to the file become effective. Tunable parameters are not stored in the.cfg files, so you need not copy the files after changing these parameters.

Softek Replicator consults the primary system configuration files of started groups many times during normal system operation (for example, when the `dtcinfo` command is run, or when the PMD daemons are started with a `launchpmds`).

To ensure that all components of Softek Replicator refer to the same configuration as the dtc device driver, Softek Replicator creates a copy of the `.cfg` file when the group is started. This copy is given a `.cur` extension since it reflects the current configuration.

The .cfg files are only processed by the `dtcstart` command. All other commands reference the .cur files. The .cur files are deleted when a group is stopped and restarted, at which point they reflect any modifications made to the configuration. This allows you to make changes to the configuration of any logical group and then allow the changes to take effect at a specific time.

▶ To prepare the Pstore:

- Enter the following command:
  `dtcinit -p <pstore_device>`

  The `dtcinit` command will set the specified device to 0.

NOTE If the specified Pstore device is used by currently operating logical groups, or the specified Pstore device is not defined in the `.cfg` file, the initialization process for the Pstore will not be processed and the `dtcinit` command will be terminated in error.

Configuring Softek Replicator with File Systems

This section describes how to initially synchronize the primary and secondary systems.

Starting Softek Replicator

After you have installed the package, created and populated license files, run `dtcconfigtool`, and distributed configuration files to secondary systems, you can start by entering:

```
/usr/dtc/bin/dtcstart -a
```

Issuing the `dtcinfo -a` command should indicate that the dtc devices are running in PASSTHRU mode, which is the default operating mode when Softek Replicator is first installed. When you create the initial mirror, the product automatically goes into NORMAL mode.

Mounting Journaled File Systems (JFS) on dtc Devices

JFS's require a log device for journaling changes to the file system metadata. JFS's can either share a single log device, or each can use a separate log device for optimal performance. For more information on organizing JFS logs for optimal performance, please refer to the *AIX Performance Management Guide*.

The log device is used to maintain a consistent file system structure, and must be replicated using a dtc device. The device that contains the file system and the log device must be replicated in the same logical group. For example, a new journal log could be associated with a file system that will be migrated via a dtc device as described in the procedure that follows.

It is assumed that two dtc devices were created in a single dtc group using dtcconfigtool. dtc0 is a suitably sized empty partition for use as the JFS log. Dtc1 is associated with an existing JFS file system that is currently unmounted.

1. To format a device for use as a JFS log, issue the following command:

   logform /dev/dtc/lg0/dsk/dtc0

2. To associate this new log with the existing file system, issue the following command:
   `chfs -a log=/dev/dtc/lg0/dsk/dtc0 /dev/dtc/lg0/rdsk/dtc1`

3. Mount the new file system as follows:
   `mount /dev/dtc/lg0/dsk/dtc1 /dtctest`

4. To configure the file system to mount automatically at boot time, modify the existing entry in the `/etc/filesystems` file:

   ```
   /dtctest:
       dev =/dev/dtc/lg0/dsk/dtc1
       vfs =jfs
       log =/dev/dtc/lg0/dsk/dtc0
       mount =true
       options =rw
       account =false
   ```

   Alternatively, you may want to create dtc devices entires in dtcconfigtool pointing to previously existing logs and file systems as local data devices. In this case, you would simply need to mount the file system using the dtc device instead of the previous native device, and modify `/etc/filesystems` as described in step 4.

How to Start Softek Replicator Mirroring

Before actively using the dtc devices, you must make an initial copy of the local data devices to the mirror devices on the secondary system.

Start Softek Replicator with the `dtcstart` command (if not already done) and create a new file system or start writing data to the data devices.

If Softek Replicator is installed on clean disks—meaning no relevant data exists on the data devices—See "Using dtcoverride" on page 25

Otherwise, if data does exist on the data devices, create the initial mirror in one of the following ways:

- See "Using launchrefresh" on page 25 to perform a full refresh operation (this is the preferred method); or
- See "Using the Courier Transport Method" on page 26.

Using dtcoverride

- Use the override command to transition to NORMAL mode. The PMDs automatically start.
  ```
  dtcoverride -a state normal
  ```

Using launchrefresh

▶ **To initiate a full refresh operation:**

1. On the primary system, type the following command:

   ```
   launchrefresh -fa
   ```

   This command synchronizes all mirror devices by transferring every data block on the primary's local data devices.

2. You can start using the dtc devices for active I/O while the refresh synchronization is taking place, without worrying about data integrity. If either system halts during a refresh operation, the process restarts at reboot.

3. You can monitor the refresh progress with the `dtcinfo` command, the `dtcmonitortool` command, or the `dtcperftool` performance monitoring utility.

NOTE Data on the mirror devices is in an incoherent, non-usable state until the refresh operation completes.

4. Once the refresh operation is complete, Softek Replicator goes to NORMAL mode and assumes normal operations.

Using the Courier Transport Method

This method is most effective when there is a tremendous amount of data to be synchronized and a full refresh operation would take too long. Follow these steps:

1. On the primary system, place Softek Replicator into TRACKING mode with the following command:

   ```
   dtcoverride -a state tracking
   ```

   At this time, you can start applications. All changes are being tracked and will eventually be copied to the secondary system.

2. On the secondary system, type the command:
   ```
   dtcrmdreco -a
   ```

3. Make a disk image backup tape of the local data devices on the primary system.

4. Transport the tape to the secondary system location, and restore the backup on the corresponding mirror devices.

5. Once the data has been applied to the mirror devices, type the following command on the secondary system:
   ```
   dtcrmdreco -ad
   ```

6. Initiate a *smart refresh* operation that goes out of TRACKING mode and transfers only the changed data since the time when the backup to tape was made and applied to the mirror devices:
   ```
   launchrefresh -a
   ```

7. After the data was transferred to the mirror devices while the system was in TRACKING mode, Softek Replicator goes to NORMAL mode and assumes typical operations.

Configuration for HP-UX

Configuring Softek Replicator

Use `dtcconfigtool` to create Softek Replicator configuration files Run it on the primary system only. To start the configuration tool, enter the following at the system prompt:

```
/opt/SFTKdtc/bin/dtcconfigtool
```

The first time `dtcconfigtool` is run, it displays an informational message shown in the following figure that explains the BAB needs to be defined.

HP-UX Configuration - dtcconfigtool Introductory Message

Select **Ok** to dismiss this message and then take the following steps:

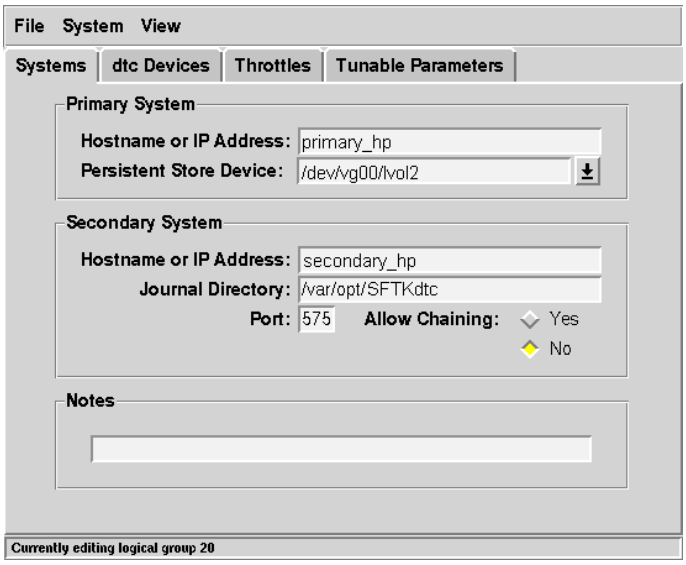
1. Allocate memory for the BAB by using the up and down arrows, or simply position the cursor in the box and enter a new value.

HP-UX Configuration - Defining the BAB

2. Select **Ok**.

`dtcconfigtool` opens the following dialog box.

HP-UX Configuration - Defining the Primary and Secondary Systems



How to Define a Logical Group

`dtcconfigtool` creates configuration files (.cfg) for each logical group. When you initially start the tool, you automatically edit Logical Group 0.

▶ To create a Logical Group other than 0:

1. From the **File** menu, select **New Logical Group**.
2. Use the arrows to select the Logical Group number for editing.
3. Under **Primary System**, the primary system **Hostname or IP Address** is filled in automatically with the name of the machine you are running on. Accept this, or enter a valid system name or IP address in this field.
4. Define the **Persistent Store Device** for the group. You can define a unique pstore for each logical group. Make sure that the pstore partition is clean and available.
5. Under **Secondary System**, specify either a resolvable system name or IP address for the secondary or mirror system.
6. To configure a local loopback configuration (that is, where both the primary and secondary are on the same system) by specifying `localhost` or `127.0.0.1` for the primary and secondary systems.
7. Keep in mind that each logical group has an independent connection, so you can define a different secondary system for each group.
8. If you specify a port number other than the default of 575 on the secondary system, you must enter that number in the **Port** field. This is the port you are connecting to on the secondary system. To change this port number, see *How to Change the Secondary System Master Daemon Port Number* on page 53.

You can also change the port that the Softek Replicator master daemon listens on as described in *How to Change the Primary System Master Daemon Port Number* on page 52.

NOTE You can define logical groups that connect to different secondary systems with different port numbers. Use this field to specify those secondary port numbers.

9. In **Journal Directory**, specify any writable directory on the secondary system for journal files.

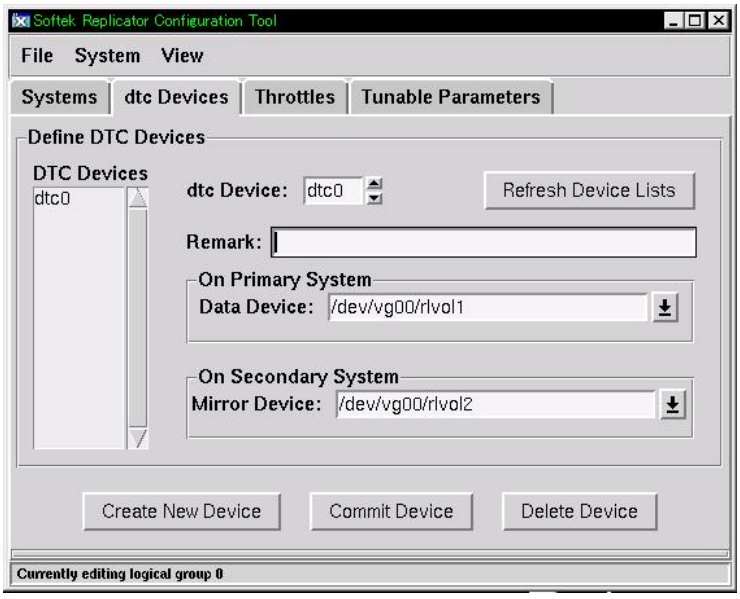
   After you complete the configuration, make sure you specify this directory in the system so that the configuration file is installed automatically at boot up. See *How to Modify /etc/fstab* on page 32.

10. If needed, change the **Allow Chaining** option (**No** is the default) for each secondary system defined. For more information on chaining, refer to *Chaining* on page 127.

How to Define dtc Devices

1. Select the **dtc Devices** tab, shown in the following figure.

HP-UX Configuration - Defining the dtc Devices



2. Device naming starts by default with dtc0. You are not required to name dtc devices sequentially or start device names with 0, but you must use the dtc# naming convention. Name the dtc device using the scrolling arrows to select the desired number or simply position the cursor in the field and type the name.

NOTE Dtc device names are automatically incremented by one when a new device is created, but you can change these names. Dtc device names must be unique only within a specific logical group. Multiple logical groups can contain identical dtc device names.

3. Choose a data device from the drop-down menu.

   You can see volumes and all started dtc devices in the data device and mirror device drop-down menus.

4. Choose a mirror device from the drop-down menu.

NOTE Each mirror device must be at least as large as its corresponding local data device.

5. Once you have defined local data and mirror devices, select **Commit Device**. If the device is already mounted, you will be asked to "Force Commit". If you do this, remember that the device needs to be unmounted (with the `umount` command) and remounted (with the `mount` command) using the dtc device, see *Configuring Softek Replicator with File Systems* on page 32.

6. To add more dtc devices to the current logical group, select **Create New Device**.

7. Repeat these steps to create all logical groups and dtc devices, then exit `dtcconfigtool`. As you exit each logical group, you are prompted to save the group definition.

NOTE Softek Replicator does not replicate a device's physical sector 0 because doing so may overwrite the device's volume table of contents, which contains critical disk label information.

How to Define Throttles

Throttles are an optional part of the Softek Replicator configuration. For instructions on creating throttles, see *Creating Throttles* on page 59 and *Throttle Reference* on page 135.

Tunable Parameters

Tunable parameters are an optional part of the configuration. To define these parameters now, see *How to Define Tunable Parameters* on page 51.

Primary and Secondary System Master Daemon Port Numbers

The default port number that the master daemon (in.dtc) uses to listen is 575. You can change this port on the primary system. Remember that the port number on the secondary system needs to match the one on the primary system. If you wish to change the port number now, see *How to Change the Primary System Master Daemon Port Number* on page 52, and *How to Change the Secondary System Master Daemon Port Number* on page 53.

Distributing the Configuration Files

The configuration files must reside on both the primary and secondary systems, since each logical group contains definitions for dtc devices, including mirror devices. Softek Replicator uses the following naming convention for these files:

`[p|s]###.cfg` – where `###` indicates the logical group number. The **p** and **s** indicate the primary and secondary systems.

You must copy the p###.cfg files onto all relevant secondary systems and rename them to s###.cfg.

1. Copy these files (using **ftp** or **rcp**) from the /etc/opt/SFTKdtc directory on primary to the same directory on the secondary system,
2. Rename the file on the secondary system, replacing the **p** with an **s**.

   For example:

   `/etc/opt/SFTKdtc/p000.cfg` (primary)

   is copied and renamed to

   `/etc/opt/SFTKdtc/s000.cfg` (secondary)

Since all logical group configuration files are stored in the same directory, the renaming allows each system in Softek Replicator to operate as both primary and secondary system without conflict.

For more information on advanced configuration options, refer to *Complex Configurations* on page 119.

NOTE If you change any primary configuration file, you must copy the updated file to the secondary system. Copy the file, then stop and restart the logical group. The `dtcstart` command processes the .cfg files, at which point changes to the file become effective. Tunable parameters are not stored in the .cfg files, so you need not copy the files after changing these parameters.

Softek Replicator consults the primary system configuration files of started groups many times during normal system operation (for example, when the `dtcinfo` command is run, or when the PMD daemons are started with a `launchpmds`).

To ensure that all components of Softek Replicator refer to the same configuration as the dtc device driver, Softek Replicator creates a copy of the `.cfg` file when the group is started. This copy is given a `.cur` extension since it reflects the current configuration.

The .cfg files are only processed by the `dtcstart` command. All other commands reference the .cur files. The .cur files are deleted when a group is stopped and restarted, at which point they reflect any modifications made to the configuration. This allows you to make changes to the configuration of any logical group and then allow the changes to take effect at a specific time.

▶ To prepare the Pstore:

- Enter the following command:
  `dtcinit -p <pstore_device>`

  The `dtcinit` command sets the specified device to 0.

NOTE If the specified Pstore device is used by currently operating logical groups, or the specified Pstore device is not defined in the `.cfg` file, the initialization process for the Pstore will not be processed and the `dtcinit` command will be terminated in error.

Configuring Softek Replicator with File Systems

This section describes how to initially synchronize the primary and secondary systems.

Starting Softek Replicator

After you have installed the package, created and populated license files, run `dtcconfigtool`, distributed configuration files to secondary systems, enter:

```
/opt/SFTKdtc/bin/dtcstart -a
```

The `dtcinfo -a` command indicates that the dtc devices are running in PASSTHRU mode, which is the default operating mode when Softek Replicator is first installed. When you create the initial mirror, the product automatically goes into NORMAL mode.

How to Modify /etc/fstab

You must modify the /etc/fstab system mounting file so that dtc devices are installed rather than the original local data devices if you are mirroring file systems across the network.

NOTE You must also add an entry to this file so that your journal file directory is mounted automatically when you boot the system.

1. If the file system to be mounted on the dtc device was just created, simply add it to the fstab file. The following example illustrates:

   ```
   /dev/dtc/lg0/dsk/dtc0 /dtctest hfs defaults 0 2
   ```

2. If the file system already exists, modify the /etc/fstab file system mounting configuration file to refer to the new dtc devices rather than the original local data devices.

   A line in *fstab* can be changed as the following example illustrates:

   If you need to create or overwrite a file system, follow the next 3 steps, otherwise skip to step 6.

   File systems can be created or overwritten before being mounted on dtc devices. This operation is straightforward when done on a dtc device in Async mode. For example, a file system can be created in the following manner:

3. Type `newfs /dev/dtc/lg0/rdsk/dtc0.`

4. When `newfs` has completed, create a mount point if one does not already exist as follows:

   ```
   mkdir /dtctest
   ```

5. Mount the new file system as follows:
   ```
   mount /dev/dtc/lg0/dsk/dtc0 /dtctest
   ```

NOTE Update the /etc/fstab file if you want the file system mounted automatically after every primary system reboot.

With VxFS, use the `mkfs` command as follows:

```
mkfs -F vxfs /dev/dtc/lg0/rdsk/dtc0
mount -F vxfs /dev/dtc/lg0/dsk/dtc0 /dtctest
```

Running a logical group in sync mode with a UFS file system does not provide true synchronous updates of the Softek Replicator devices due to the asynchronous ordering of the disk writes by the UFS file system.

When running a logical group in sync mode with VxFS mounted use the `mincache=dsync` option to ensure that the underlying VxFS does not cache the writes but instead writes them directly to disk:

```
$ mount -F vxfs -o mincache=dsync /dev/dtc/lg0/dsk/dtc0 /dtctest
```

NOTE Performance may be noticeably degraded when running in Sync mode.

6. Unmount (with the `umount` command) and remount (with the `mount` or `mountall` command) the affected file systems before changing their contents.

Databases and other applications can work directly with raw disk devices rather than using file systems for data storage. These application configurations must be changed to refer to the new Softek Replicator devices rather than the local data devices.

Creating devices does not modify the original contents of the local data devices. It is appropriate to define dtc devices on top of pre-existing data. No export or import of data is required.

NOTE **Error Messages:**
When dtc devices are described in /etc/fstab file, a mount device error will occur at system restart. However, an auto-mount process will be launched for devices that couldn't be mounted.

If an application that uses dtc device is set to auto-start at system restart, please confirm the starting timing for the application. If the application auto-start is registered in `rc` script before `rc2.d`, please re-reregister it in `rc` script after `rc2.d`.

How to Start Softek Replicator Mirroring

Before actively using the dtc devices, you must make an initial copy of the local data devices to the mirror devices on the secondary system.

Start Softek Replicator with the `dtcstart` command (if not already done) and create a new file system or start writing data to the data devices.

If Softek Replicator is installed on clean disks—meaning no relevant data exists on the data devices—then use the following method:

- *Using dtcoverride.*

Otherwise, if data does exist on the data devices, create the initial mirror in one of the following ways:

- *Using launchrefresh* to perform a full refresh operation (this is the preferred method).
- *Using the Courier Transport Method.*

Using dtcoverride

▶ **To initiate mirroring:**

- Use the override command to transition to NORMAL mode. The PMDs automatically start.
  `dtcoverride -a state normal`

Using launchrefresh

▶ **To initiate a full refresh operation:**

1. On the primary system, type the following command:

   `launchrefresh -fa`

   This command synchronizes all mirror devices by transferring every data block on the primary's local data devices.

2. You can start using the dtc devices for active I/O while the refresh synchronization is taking place, without worrying about data integrity. If either system halts during a refresh operation, the process restarts at reboot.

3. You can monitor the refresh progress with the `dtcinfo` command, the `dtcmonitortool` command, or the `dtcperftool` performance monitoring utility.

NOTE Data on the mirror devices is in an incoherent, non-usable state until the refresh operation completes.

4. Once the refresh operation is complete, Softek Replicator goes to NORMAL mode and assumes normal operations.

Using the Courier Transport Method

This method is most effective when there is a tremendous amount of data to be synchronized and a full refresh operation would take too long. Follow these steps:

1. On the primary system, place Softek Replicator into TRACKING mode with the following command:

   `dtcoverride -a state tracking`

   At this time, you can start applications. All changes are being tracked and will eventually be copied to the secondary system.

2. On the secondary system, type the command:
   `dtcrmdreco -a`

3. Make a disk image backup tape of the local data devices on the primary system.

4. Transport the tape to the secondary system location, and restore the backup on the corresponding mirror devices.

5. Once the data has been applied to the mirror devices, type the following command on the secondary system:
   `dtcrmdreco -ad`

6. Initiate a *smart refresh* operation that goes out of TRACKING mode and transfers only the changed data since the time when the backup to tape was made and applied to the mirror devices:
   `launchrefresh -a`

7. After the data was transferred to the mirror devices while the system was in TRACKING mode, Softek Replicator goes to NORMAL mode and assumes typical operations.

Configuration for Linux

Configuring Softek Replicator

Use `dtcconfigtool` to create Softek Replicator configuration files Run it on the primary system only. To start the configuration tool, enter the following at the system prompt:

```
/opt/SFTKdtc/bin/dtcconfigtool
```

The first time `dtcconfigtool` is run, it displays an informational message shown in the following figure that explains the BAB needs to be defined.

Linux Configuration - dtcconfigtool Introductory Message

Select **Ok** to dismiss this message and then take the following steps:

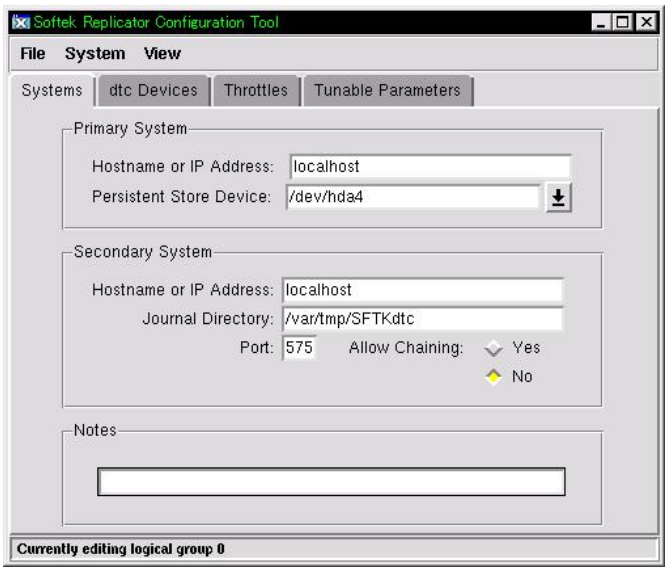
1. Allocate memory for the BAB by using the up and down arrows, or simply position the cursor in the box and enter a new value.

Linux Configuration - Defining the BAB

2. Select **Ok**.

   `dtcconfigtool` opens the following dialog box.

Linux Configuration - Defining the Primary and Secondary Systems



How to Define a Logical Group

`dtcconfigtool` creates configuration files (.cfg) for each logical group. When you initially start the tool, you automatically edit Logical Group 0.

▶ To create a Logical Group other than 0:

1. From the **File** menu, select **New Logical Group**.
2. Use the arrows to select the Logical Group number for editing.
3. Under **Primary System**, the primary system **Hostname or IP Address** is filled in automatically with the name of the machine you are running on. Accept this, or enter a valid system name or IP address in this field.
4. Define the **Persistent Store Device** (**pstore**) for the group. You can define a unique pstore for each logical group.
5. Under **Secondary System**, specify either a resolvable system name or IP address for the secondary or mirror system.
6. To configure a local loopback configuration (that is, where both the primary and secondary are on the same system) by specifying `localhost` or `127.0.0.1` for the primary and secondary systems.
7. Keep in mind that each logical group has an independent connection, so you can define a different secondary system for each group.
8. If you specify a port number other than the default of 575 on the secondary system, you must enter that number in the **Port** field. This is the port you are connecting to on the secondary system. To change this port number, see *How to Change the Secondary System Master Daemon Port Number* on page 53.

You can also change the port that the Softek Replicator master daemon listens on as described in *How to Change the Primary System Master Daemon Port Number* on page 52.

NOTE You can define logical groups that connect to different secondary systems with different port numbers. Use this field to specify those secondary port numbers.

9. In **Journal Directory**, specify any writable directory on the secondary system for journal files.

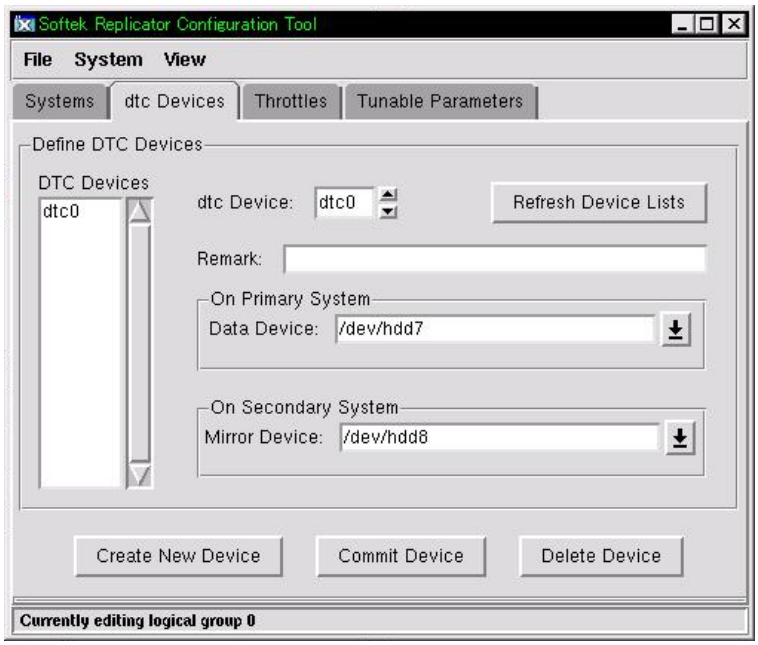
   After you complete the configuration, make sure you specify this directory in the system so that the configuration file is installed automatically at boot up. See *How to Modify /etc/fstab* on page 40.

10. If needed, change the **Allow Chaining** option (**No** is the default) for each secondary system defined. For more information on chaining, refer to *Chaining* on page 127.

How to Define dtc Devices

1. Select the **dtc Devices** tab, shown in the following figure.

Linux Configuration - Defining the dtc Devices



2. Device naming starts by default with dtc0. You are not required to name dtc devices sequentially or start device names with 0, but you must use the *dtc#* naming convention. Name the dtc device using the scrolling arrows to select the desired number or simply position the cursor in the field and type the name.

NOTE Dtc device names are automatically incremented by one when a new device is created, but you can change these names. Dtc device names must be unique only within a specific logical group. Multiple logical groups can contain identical dtc device names.

3. Choose a data device from the drop-down menu.

   You can see volumes and all started dtc devices in the data device and mirror device drop-down menus.

4. Choose a mirror device from the drop-down menu.

NOTE Each mirror device must be at least as large as its corresponding local data device.

5. Once you have defined local data and mirror devices, select **Commit Device**.

6. To add more dtc devices to the current logical group, select **Create New Device**.

7. Repeat these steps to create all logical groups and dtc devices, then exit `dtcconfigtool`. As you exit each logical group, you are prompted to save the group definition.

NOTE Softek Replicator does not replicate a device's physical sector 0 because doing so may overwrite the device's volume table of contents, which contains critical disk label information.

How to Define Throttles

Throttles are an optional part of the Softek Replicator configuration. For instructions on creating throttles, see *Administering Softek Replicator* on page 57 and *Throttle Reference* on page 135.

Tunable Parameters

Tunable parameters are an optional part of the configuration. To define these parameters now, see *How to Define Tunable Parameters* on page 51.

Primary and Secondary System Master Daemon Port Numbers

The default port number that the master daemon (in.dtc) uses to listen is 575. You can change this port on the primary system. Remember that the port number on the secondary system needs to match the one on the primary system. If you wish to change the port number now, see *How to Change the Primary System Master Daemon Port Number* on page 52, and *How to Change the Secondary System Master Daemon Port Number* on page 53.

Distributing the Configuration Files

The configuration files must reside on both the primary and secondary systems, since each logical group contains definitions for dtc devices, including mirror devices. Softek Replicator uses the following naming convention for these files:

`[p|s]###.cfg` — where ### indicates the logical group number. The `p` and `s` indicate the primary and secondary systems.

You must copy the p###.cfg files onto all relevant secondary systems and rename them to s###.cfg.

1. Copy these files (using **ftp** or **rcp**) from the `/etc/opt/SFTKdtc` directory on primary to the same directory on the secondary system,
2. Rename the file on the secondary system, replacing the `p` with an `s`.

   For example:

   `/etc/opt/SFTKdtc/p000.cfg (primary)`

   is copied and renamed to

   `/etc/opt/SFTKdtc/s000.cfg (secondary)`

Since all logical group configuration files are stored in the same directory, the renaming allows each system in Softek Replicator to operate as both primary and secondary system without conflict.

For more information on advanced configuration options, refer to *Complex Configurations* on page 119.

NOTE If you change any primary configuration file, you must copy the updated file to the secondary system. Copy the file, then stop and restart the logical group. The `dtcstart` command processes the .cfg files, at which point changes to the file become effective. Tunable parameters are not stored in the .cfg files, so you need not copy the files after changing these parameters.

Softek Replicator consults the primary system configuration files of started groups many times during normal system operation (for example, when the `dtcinfo` command is run, or when the PMD daemons are started with a `launchpmds`).

To ensure that all components of Softek Replicator refer to the same configuration as the dtc device driver, Softek Replicator creates a copy of the `.cfg` file when the group is started. This copy is given a `.cur` extension since it reflects the current configuration.

The `.cfg` files are only processed by the `dtcstart` command. All other commands reference the `.cur` files. The .cur files are deleted when a group is stopped and restarted, at which point they reflect any modifications made to the configuration. This allows you to make changes to the configuration of any logical group and then allow the changes to take effect at a specific time.

▶ To prepare the Pstore:

- Enter the following command:
  `dtcinit -p <pstore_device>`

The `dtcinit` command sets the specified device to 0.

NOTE If the specified Pstore device is used by currently operating logical groups, or the specified Pstore device is not defined in the `.cfg` file, the initialization process for the Pstore will not be processed and the `dtcinit` command will be terminated in error.

Configuring Softek Replicator with File Systems

This section describes how to initially synchronize the primary and secondary systems.

Databases and other applications can work directly with raw disk devices rather than using file systems for data storage. These application configurations must be changed to refer to the new Softek Replicator devices rather than the local data devices. Creating devices does not modify the original contents of the local data devices. It is appropriate to define dtc devices on top of pre-existing data. No export or import of data is required.

Starting Softek Replicator

After you have installed the package, created and populated license files, run `dtcconfigtool`, and distributed configuration files to secondary systems, you can start by entering:

```
/opt/SFTKdtc/bin/dtcstart -a
```

Issuing the `dtcinfo -a` command should indicate that the dtc devices are running in PASSTHRU mode, which is the default operating mode when Softek Replicator is first installed. When you create the initial mirror, the product automatically goes into NORMAL mode.

How to Modify /etc/fstab

To modify the `/etc/fstab` system mounting file so that dtc devices are installed rather than the original local data devices if you are mirroring file systems across the network, proceed as follows:

1. If the file system to be mounted on the dtc device was just created, simply add it to the `fstab` file. The following example, using an `ext2` file system, illustrates this process:

   ```
   /dev/dtc/lg0/dsk/dtc0 /dtctest ext2 defaults 0 2
   ```

2. If the file system already exists, modify the `/etc/fstab` file system mounting configuration file to refer to the new dtc devices rather than the original local data devices, as in the following example:
   ```
   /dev/hdc3 /datadev ext defaults 1 2
   ```

   to:

   ```
   /dev/dtc/lg0/dsk/dtc0 /datadev ext defaults 1 2
   ```

To create or overwrite a file system follow the next 3 steps; otherwise skip to step 6.

CAUTION:
File systems can be created or overwritten before being mounted on dtc devices. This operation is straightforward when done on a dtc device in Async mode.Performance may be noticeably degraded when running in Sync mode.

3. Type:
   `/sbin/mkfs /dev/dtc/lg0/dsk/dtc0`
4. When `mkfs` has completed, create a mount point if one does not already exist. Type:
   `mkdir /dtctest`
5. To mount the new file system, type:
   `mount /dev/dtc/lg0/dsk/dtc0 /dtctest`
6. Mount the new file system as follows:
   `mount /dev/dtc/lg0/dsk/dtc0 /dtctest`

NOTE Update the `/etc/fstab` file if you want the file system mounted automatically after every primary system reboot.

How to Start Softek Replicator Mirroring

Before actively using the dtc devices, you must make an initial copy of the local data devices to the mirror devices on the secondary system.

Start Softek Replicator with the `dtcstart` command (if not already done) and create a new file system or start writing data to the data devices.

If Softek Replicator is installed on clean disks—meaning no relevant data exists on the data devices—then use the following method:

- *Using dtcoverride*.

Otherwise, if data does exist on the data devices, create the initial mirror in one of the following ways:

- *Using launchrefresh* to perform a full refresh operation (this is the preferred method).
- *Using the Courier Transport Method*.

Using dtcoverride

▶ **To initiate mirroring:**

- Use the override command to transition to NORMAL mode. The PMDs automatically start.
  `dtcoverride -a state normal`

Using launchrefresh

▶ **To initiate a full refresh operation:**

1. On the primary system, type the following command:
   `launchrefresh -fa`
   This command synchronizes all mirror devices by transferring every data block on the primary's local data devices.
2. You can start using the dtc devices for active I/O while the refresh synchronization is taking place, without worrying about data integrity. If either system halts during a refresh operation, the process restarts at reboot.

3. You can monitor the refresh progress with the `dtcinfo` command, the `dtcmonitortool` command, or the `dtcperftool` performance monitoring utility.

NOTE Data on the mirror devices is in an incoherent, non-usable state until the refresh operation completes.

4. Once the refresh operation is complete, Softek Replicator goes to NORMAL mode and assumes normal operations.

Using the Courier Transport Method

This method is most effective when there is a tremendous amount of data to be synchronized and a full refresh operation would take too long. Follow these steps:

1. On the primary system, place Softek Replicator into TRACKING with the following command:

   ```
   dtcoverride -a state tracking
   ```

   At this time, you can start applications. All changes are being tracked and will eventually be copied to the secondary system.

2. On the secondary system, type the command:
   ```
   dtcrmdreco -a
   ```

3. Make a disk image backup tape of the local data devices on the primary system.

4. Transport the tape to the secondary system location, and restore the backup on the corresponding mirror devices.

5. Once the data has been applied to the mirror devices, type the following command on the secondary system:
   ```
   dtcrmdreco -ad
   ```

6. Initiate a *smart refresh* operation that goes out of TRACKING mode and transfers only the changed data since the time when the backup to tape was made and applied to the mirror devices:
   ```
   launchrefresh -a
   ```

7. After the data was transferred to the mirror devices while the system was in TRACKING mode, Softek Replicator goes to NORMAL mode and assumes typical operations.

Configuration for Solaris

Configuring Softek Replicator

Use `dtcconfigtool` to create Softek Replicator configuration files Run it on the primary system only. To start the configuration tool, enter the following at the system prompt:

```
/opt/SFTKdtc/bin/dtcconfigtool
```

The first time `dtcconfigtool` is run, it displays an informational message shown in the following figure that explains the BAB needs to be defined.

Solaris Configuration- dtcconfigtool Introductory Message

Select **Ok** to dismiss this message and then take the following steps:

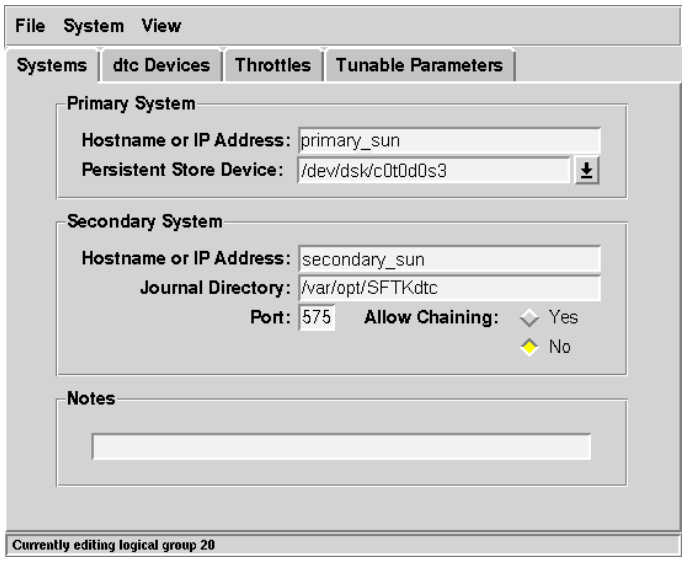
1. Allocate memory for the BAB by using the up and down arrows, or simply position the cursor in the box and enter a new value.

Solaris Configuration- Defining the BAB

2. Select **Ok**.

   `dtcconfigtool` opens the following dialog box.

Solaris Configuration - Defining the Primary and Secondary Systems



How to Define a Logical Group

`dtcconfigtool` creates configuration files (.cfg) for each logical group. When you initially start the tool, you automatically edit Logical Group 0.

▶ **To create a Logical Group other than 0:**

1. From the **File** menu, select **New Logical Group**.
2. Use the arrows to select the Logical Group number for editing.
3. Under **Primary System**, the primary system **Hostname or IP Address** is filled in automatically with the name of the machine you are running on. Accept this, or enter a valid system name or IP address in this field.
4. Define the **Persistent Store Device** (**pstore**) for the group. You can define a unique pstore for each logical group.
5. Under **Secondary System**, specify either a resolvable system name or IP address for the secondary or mirror system.
6. To configure a local loopback configuration (that is, where both the primary and secondary are on the same system) by specifying `localhost` or `127.0.0.1` for the primary and secondary systems.
7. Keep in mind that each logical group has an independent connection, so you can define a different secondary system for each group.
8. If you specify a port number other than the default of 575 on the secondary system, you must enter that number in the **Port** field. This is the port you are connecting to on the secondary system. To change this port number, see *How to Change the Secondary System Master Daemon Port Number* on page 53.

You can also change the port that the Softek Replicator master daemon listens on as described in *How to Change the Primary System Master Daemon Port Number* on page 52.

NOTE You can define logical groups that connect to different secondary systems with different port numbers. Use this field to specify those secondary port numbers.

9. In **Journal Directory**, specify any writable directory on the secondary system for journal files.

   After you complete the configuration, make sure you specify this directory in the system so that the configuration file is installed automatically at boot up. See *How to Modify /etc/vfstab* on page 48.

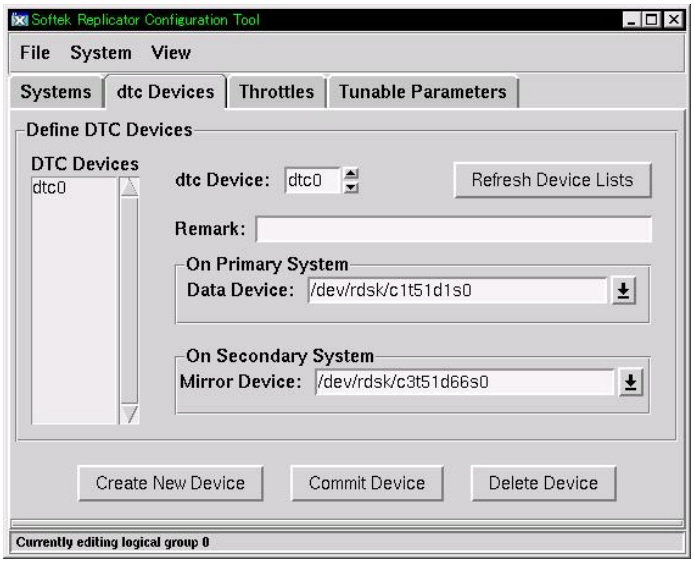
10. If needed, change the **Chaining** option (**No** is the default) for each secondary system defined. For more information on chaining, refer to *Chaining* on page 127.

CAUTION:
Softek Replicator will generate errors during the replication process if you are using primary or secondary Solaris Solstice Disk Suite (SDS) dtc devices, with partitions (in either concat or striped mode) starting at cylinder 0. Your SDS partitions must start at cylinder 1, or greater.

How to Define dtc Devices

1. Select the **dtc Devices** tab, shown in the following figure.

Solaris Configuration - Defining the dtc Devices



2. Device naming starts by default with dtc0. You are not required to name dtc devices sequentially or start device names with 0, but you must use the *dtc#* naming convention. Name the dtc device using the scrolling arrows to select the desired number or simply position the cursor in the field and type the name.

NOTE Dtc device names are automatically incremented by one when a new device is created, but you can change these names. Dtc device names must be unique only within a specific logical group. Multiple logical groups can contain identical dtc device names.

3. Choose a data device from the drop-down menu.

   You can see volumes and all started dtc devices in the data device and mirror device drop-down menus.

4. Choose a mirror device from the drop-down menu.

NOTE Each mirror device must be at least as large as its corresponding local data device.

5. Once you have defined local data and mirror devices, select **Commit Device**.
6. To add more dtc devices to the current logical group, select **Create New Device**.
7. Repeat these steps to create all logical groups and dtc devices, then exit `dtcconfigtool`. As you exit each logical group, you are prompted to save the group definition.

NOTE Softek Replicator does not replicate a device's physical sector 0 because doing so may overwrite the device's volume table of contents, which contains critical disk label information.

How to Define Throttles

Throttles are an optional part of the Softek Replicator configuration. For instructions on creating throttles, see *Administering Softek Replicator* on page 57.

Tunable Parameters

Tunable parameters are an optional part of the configuration. To define these parameters now, see *How to Define Tunable Parameters* on page 51.

Primary and Secondary System Master Daemon Port Numbers

The default port number that the master daemon (in.dtc) uses to listen is 575. You can change this port on the primary system. Remember that the port number on the secondary system needs to match the one on the primary system. If you wish to change the port number now, see *How to Change the Primary System Master Daemon Port Number* on page 52, and *How to Change the Secondary System Master Daemon Port Number* on page 53.

Distributing the Configuration Files

The configuration files must reside on both the primary and secondary systems, since each logical group contains definitions for dtc devices, including mirror devices. Softek Replicator uses the following naming convention for these files:

`[p|s]###.cfg` — where ### indicates the logical group number. The `p` and `s` indicate the primary and secondary systems.

You must copy the p###.cfg files onto all relevant secondary systems and rename them to s###.cfg.

1. Copy these files (using **ftp** or **rcp**) from the /etc/opt/SFTKdtc directory on primary to the same directory on the secondary system,
2. Rename the file on the secondary system, replacing the `p` with an `s`.

   For example:

   `/etc/opt/SFTKdtc/p000.cfg (primary)`

   is copied and renamed to

   `/etc/opt/SFTKdtc/s000.cfg (secondary)`

Since all logical group configuration files are stored in the same directory, the renaming allows each system in Softek Replicator to operate as both primary and secondary system without conflict.

For more information on advanced configuration options, refer to *Complex Configurations* on page 119.

NOTE If you change any primary configuration file, you must copy the updated file to the secondary system. Copy the file, then stop and restart the logical group. The `dtcstart` command processes the `.cfg` files, at which point changes to the file become effective. Tunable parameters are not stored in the `.cfg` files, so you need not copy the files after changing these parameters.

Softek Replicator consults the primary system configuration files of started groups many times during normal system operation (for example, when the `dtcinfo` command is run, or when the PMD daemons are started with a `launchpmds`).

To ensure that all components of Softek Replicator refer to the same configuration as the dtc device driver, Softek Replicator creates a copy of the `.cfg` file when the group is started. This copy is given a `.cur` extension since it reflects the current configuration.

The `.cfg` files are only processed by the `dtcstart` command. All other commands reference the `.cur` files. The `.cur` files are deleted when a group is stopped and restarted, at which point they reflect any modifications made to the configuration. This allows you to make changes to the configuration of any logical group and then allow the changes to take effect at a specific time.

▶ To prepare the Pstore:

- Enter the following command:
  `dtcinit -p <pstore_device>`

The `dtcinit` command sets the specified device to 0.

NOTE If the specified Pstore device is used by currently operating logical groups, or the specified Pstore device is not defined in the `.cfg` file, the initialization process for the Pstore will not be processed and the `dtcinit` command will be terminated in error.

Configuring Softek Replicator with File Systems

This section describes how to initially synchronize the primary and secondary systems.

Starting Softek Replicator

After you have installed the package, created and populated license files, run `dtcconfigtool`, and distributed configuration files to secondary systems, you can start by entering:

```
/opt/SFTKdtc/bin/dtcstart -a
```

Issuing the `dtcinfo -a` command should indicate that the dtc devices are running in PASSTHRU mode, which is the default operating mode when Softek Replicator is first installed. When you create the initial mirror, the product automatically goes into NORMAL mode.

How to Modify /etc/vfstab

You must modify the /etc/vfstab system mounting file so that dtc devices are installed rather than the original local data devices if you are mirroring file systems across the network.

NOTE You must also add an entry to this file so that your journal file directory is mounted automatically when you boot the system.

1. If the file system to be mounted on the dtc device was just created, simply add it to the fstab file. The following example illustrates:

   ```
   /dev/dtc/lg0/dsk/dtc0 /dtctest hfs defaults 0 2
   ```

2. Modify the */etc/vfstab* file system mounting configuration file to refer to the new dtc devices rather than the original local data devices.

   If you need to create or overwrite a file system, follow the next 3 steps, otherwise skip to step 6.

   File systems can be created or overwritten before being mounted on **dtc** devices. This operation is straightforward when done on a dtc device in Async mode. For example, a file system can be created in the following manner:

3. Type `newfs /dev/dtc/lg0/rdsk/dtc0.`

NOTE If you are using Solaris 8, type `newfs` using the actual physical raw device: `newfs /dev/rdsk/c0t0d0s4.`

4. When `newfs` has completed, create a mount point if one does not already exist as follows:

   ```
   mkdir /dtctest
   ```

5. Mount the new file system as follows:

```
mount /dev/dtc/lg0/dsk/dtc0 /dtctest
```

NOTE Update the /etc/vfstab file if you want the file system mounted automatically after every primary system reboot.

With `vxfs`, use the `mkfs` command as follows:

```
mkfs -F vxfs /dev/dtc/lg0/rdsk/dtc0
mount -F vxfs /dev/dtc/lg0/dsk/dtc0 /dtctest
```

NOTE If you are using Solaris 8, type `mkfs` using the actual physical raw device:
`mkfs -F vxfs /dev/rdsk/c0t0d0s4`

Running a logical group in sync mode with a UFS file system does not provide true synchronous updates of the Softek Replicator devices due to the asynchronous ordering of the disk writes by the UFS file system.

When running a logical group in sync mode with VxFS mounted use the `mincache=dsync` option to ensure that the underlying VxFS does not cache the writes but instead writes them directly to disk:

```
$ mount -F vxfs -o mincache=dsync /dev/dtc/lg0/dsk/dtc0 /dtctest
```

NOTE Performance may be noticeably degraded when running in Sync mode.

6. Unmount (with the `umount` command) and remount (with the `mount` or `mountall` command) the affected file systems before changing their contents.

Databases and other applications can work directly with raw disk devices rather than using file systems for data storage. These application configurations must be changed to refer to the new Softek Replicator devices rather than the local data devices. Creating devices does not modify the original contents of the local data devices. It is appropriate to define dtc devices on top of pre-existing data. No export or import of data is required.

How to Start Softek Replicator Mirroring

Before actively using the dtc devices, you must make an initial copy of the local data devices to the mirror devices on the secondary system.

Start Softek Replicator with the `dtcstart` command (if not already done) and create a new file system or start writing data to the data devices.

If Softek Replicator is installed on clean disks—meaning no relevant data exists on the data devices—then use the following method:

- *Using dtcoverride*.

Otherwise, if data does exist on the data devices, create the initial mirror in one of the following ways:

- *Using launchrefresh* to perform a full refresh operation (this is the preferred method).
- *Using the Courier Transport Method*.

Using dtcoverride

▶ **To initiate mirroring:**

- Use the override command to transition to NORMAL mode. The PMDs automatically start.
  `dtcoverride -a state normal`

Using launchrefresh

▶ **To initiate a full refresh operation:**

1. On the primary system, type the following command:

   `launchrefresh -fa`

   This command synchronizes all mirror devices by transferring every data block on the primary's local data devices.

2. You can start using the dtc devices for active I/O while the refresh synchronization is taking place, without worrying about data integrity. If either system halts during a refresh operation, the process restarts at reboot.

3. You can monitor the refresh progress with the `dtcinfo` command, the `dtcmonitortool` command, or the `dtcperftool` performance monitoring utility.

NOTE Data on the mirror devices is in an incoherent, non-usable state until the refresh operation completes.

4. Once the refresh operation is complete, Softek Replicator goes to NORMAL mode and assumes normal operations.

Using the Courier Transport Method

This method is most effective when there is a tremendous amount of data to be synchronized and a full refresh operation would take too long. Follow these steps:

1. On the primary system, place Softek Replicator into TRACKING mode with the following command:

   `dtcoverride -a state tracking`

   At this time, you can start applications. All changes are being tracked and will eventually be copied to the secondary system.

2. On the secondary system, type the command:
   `dtcrmdreco -a`

3. Make a disk image backup tape of the local data devices on the primary system.

4. Transport the tape to the secondary system location, and restore the backup on the corresponding mirror devices.

5. Once the data has been applied to the mirror devices, type the following command on the secondary system:
   `dtcrmdreco -ad`

6. Initiate a *smart refresh* operation that goes out of TRACKING mode and transfers only the changed data since the time when the backup to tape was made and applied to the mirror devices:
   `launchrefresh -a`

7. After the data was transferred to the mirror devices while the system was in TRACKING mode, Softek Replicator goes to NORMAL mode and assumes typical operations.

Common Configuration for AIX, HP-UX, Linux and Solaris

How to Define Tunable Parameters

Tunable parameters are an optional part of the configuration. If you choose not to define these parameters, the defaults in the following table are used.

Tunable Parameters

CHUNKDELAY: 0 milliseconds (Minimum 0; Maximum 999)	CHUNKSIZE: 2048 KB (Minimum 64; Maximum 16384)
SYNCMODE: OFF	SYNCMODEDEPTH: 1 entry (Minimum 1; Maximum 2147483647)
SYNCMODETIMEOUT: 30 seconds (Minimum 1; Maximum 86400)	NETMAXKBPS: -1 KB (-1 indicates off) (Minimum 1; Maximum 2147483647)
STATINTERVAL: 10 seconds (Minimum 1; Maximum 86400)	MAXSTATFILESIZE: 64 KB (Minimum 1; Maximum 32000)
COMPRESSION: OFF	TRACETHROTTLE: OFF

You can set tunable parameters during initial configuration or later, with the `dtcset` command. Tunable parameter changes become effective within STATINTERVAL seconds (default is 10).

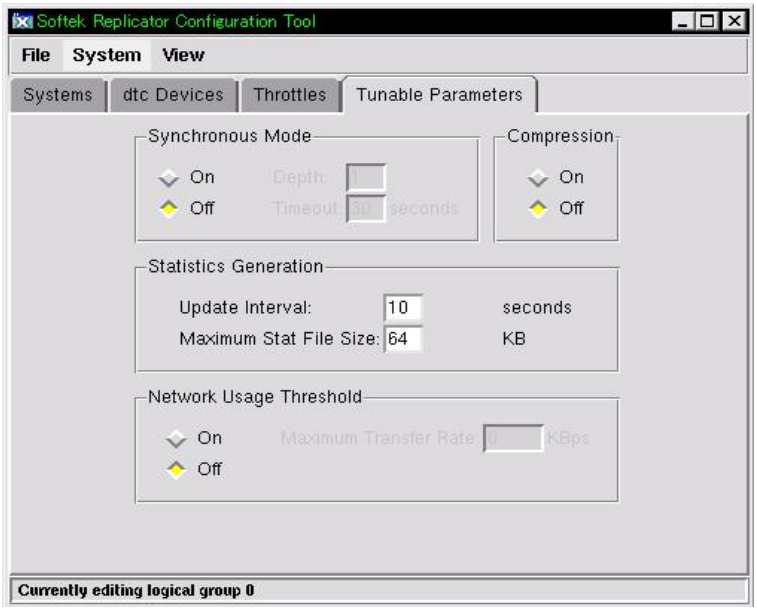
▶ **To change tunable parameters using the Configuration Tool:**

1. From the **File** menu, select the logical group whose parameters you want to change.

2. Select the **Tunable Parameters** tab.

   See figure: *Configuring AIX, HP-UX, Linux and Solaris - Tunable Parameters Tab in dtcconfigtool* on page 52.

Configuring AIX, HP-UX, Linux and Solaris - Tunable Parameters Tab in dtcconfigtool



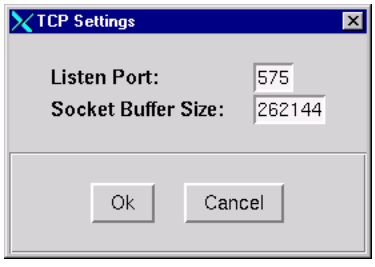
3. Edit the parameter definitions as needed. For a description of all parameters, see *Tunable Parameters* on page 143.
4. From the **File** menu, **Save Changes**.
5. Run `dtcstart`.

For information on editing throttles and tunable parameters after the initial configuration, refer to *Creating Throttles* on page 59 and *Throttle Reference* on page 135.

How to Change the Primary System Master Daemon Port Number

1. First, make sure that the port number you want to use is available.
2. From the **System** menu, select **TCP Settings**.

Configuring AIX. HP-UX, Linux and Solaris - TCP Settings



3. In **Listen Port**, enter the port number you want the Softek Replicator master daemon to listen to on the primary system.

4. If needed, you can also change the **Socket Buffer Size**. This parameter sets the TCP/IP send and receive buffer size in bytes.

NOTE The secondary system must have the same port number as the primary system. If you change the port number on the primary system, you must also change it on the secondary one.

5. Exit the Configuration Tool.
6. Remember to copy the configuration file (p000.cfg) from the primary to the secondary system as described in *Distributing the Configuration Files* on page 30.
7. Type `dtcstop` followed by `dtcstart` so that the new configuration file takes effect.
8. Type `killdtcmaster`, followed by `launchdtcmaster` to restart the master daemon.

How to Change the Secondary System Master Daemon Port Number

If you change the port number on the primary system, you must change it on the secondary system.

▶ **To change the port number on the secondary system:**

1. In, locate the line with **in.dtc** in it. For example:

```
in.dtc                  575/TCP
```

2. Edit the port number so that it matches the port number you specified on the primary system.
3. Save the file.
4. Type `killdtcmaster`, followed by `launchdtcmaster` to restart the master daemon.

Configuring Softek Replicator for Relational Database Management Systems (RDBMS)

Many production database environments install the database on top of raw disk partitions rather than on top of file systems. These raw disk partitions may actually be volumes created by either VxVM or volume managers.

When the database is created, it stores the paths to these raw disk devices as part of the database itself. This is a challenge for Softek Replicator, or for any disaster recovery scheme that needs to restore the database on another computer whose disk layout may not be identical to the disk layout of the original machine.

The solution is to specify a symbolic link path rather than the actual path to the raw disk or volume when creating the database. The method is illustrated in the following example.

Suppose you have created a database on three raw disk devices that are actually striped volumes. The normal specification to the RDBMS software would be:

```
TABLESPACE A: /dev/vx/rdsk/sybasedg/vol01
TABLESPACE B: /dev/vx/rdsk/sybasedg/vol02
TABLESPACE C: /dev/vx/rdsk/sybasedg/vol03
```

Sybase is used in this example, but this applies to Oracle or other RDBMS packages as well. Instead, use the following procedure:

1. Create symbolic link definitions to these volumes and give those paths to the database.

```
$ mkdir /dev/devlinks
$ ln -s /dev/vx/rdsk/sybasedg/vol01 \ /dev/devlinks/tblspA
$ ln -s /dev/vx/rdsk/sybasedg/vol02 \ /dev/devlinks/tblspB
$ ln -s /dev/vx/rdsk/sybasedg/vol03 \ /dev/devlinks/tblspC
TABLESPACE A: /dev/devlinks/tblspA
TABLESPACE B: /dev/devlinks/tblspB
TABLESPACE C: /dev/devlinks/tblspC
```

2. Create and populate the database as you would normally.
3. Shut down the database.
4. Install Softek Replicator and configure it to mirror these volumes; change the symbolic links to point to the created dtc devices instead of the original raw disks or volumes.
5. Follow the usual steps for creating dtc devices. Do not start the PMDs yet.
6. Create the symbolic links:

```
$rm /dev/devlinks/tblspA
$ln -s /dev/dtc/lg0/rdsk/dtc0 \ /dev/devlinks/tblspA
$rm /dev/devlinks/tblspB
$ln -s /dev/dtc/lg0/rdsk/dtc1 \ /dev/devlinks/tblspB
$rm /dev/devlinks/tblspC
$ln -s /dev/dtc/lg0/rdsk/dtc2 \ /dev/devlinks/tblspC
```

NOTE The sample script is provided in order to change symbolic links. Please see Appendix D for a detailed explanation.

7. Type `launchrefresh -af`.
8. Monitor for the completion of the refresh operation with `dtcmonitortool` or `dtcperftool`.
9. Most RDBMS must own the raw device against which they perform I/O. Therefore, you must change the ownership of the actual dtc devices involved as follows:

```
$ killpmds
$ chown -R sybase:sybase /dev/dtc/lg0
$ launchpmds
```

10. Now you can bring up the database so that it accesses its data through the dtc devices, and all updates are mirrored to the secondary system

> **CAUTION:**
> Most RDBMS systems configured for client/server operations capture system identification for the RDBMS server in configuration files or in the database itself. Attempting to bring up the RDBMS on the secondary system after a switch-over without correcting this server identification results in warnings or fatal errors. A RDBMS should never be run on a secondary system without a legal right to do so under the licensing terms from the database vendor.

Moving a Non-Symbolic Linked Database to Softek Replicator

You may encounter a situation in which a legacy database uses the absolute path to the raw disk devices.

There are two ways to deal with this situation:

1. Entirely recreate the database as follows:

   Export the database to tape.

   Create a new database on top of the symbolic links.

   Import the database back from tape to the new database.

2. Redirect the database through symbolic links. (This is risky and may adversely affect volume managers!)

   Change the names of the raw devices or volumes used.

   Create symbolic links from the old device names to the new paths.

   Follow the steps described in *Configuring Softek Replicator for Relational Database Management Systems (RDBMS)* on page 53.

3

Administering Softek Replicator

Creating Throttles 59

Changing Tunable Parameters 63

Adding/Modifying Devices 65

Deleting Logical Groups 67

Relocating the pstore 68

Modifying the BAB 68

Resizing the Journal File Directory 69

Checkpointing Softek Replicator 70

Mounting Mirror Devices on HP-UX and AIX 77

Rebooting the Primary System on HP-UX 77

This chapter covers all Softek Replicator administration tasks that are performed after the initial installation and configuration of the software.

Creating Throttles

A throttle is a set of statements you can define to optimize Softek Replicator, Open Systems Edition, by monitoring elements, measuring variables, and performing actions based on evaluations. Throttles can be defined during the initial configuration of Softek Replicator and they can be edited at any point during operations. Throttles are defined on a logical group basis and the definitions are stored in the `.cfg` files. For more information on throttles, see *Throttle Reference* on page 135.

Editing Throttle Definitions

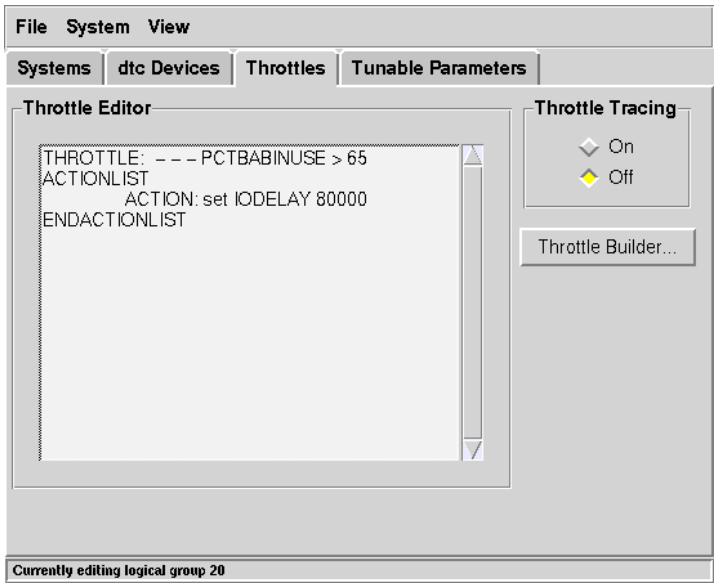
- Determine a timeframe for the throttles to be active.
- Construct an expression that identifies the variable to be monitored by the throttle and establishes the parameters for that variable.
- The Configuration Tool provides drop-down menus with lists of options for these sections.
- Define actions for the throttle to perform when variable measurements fall outside the defined parameters.

Refer to *Throttle Examples* on page 62.

▶ To define a throttle:

1. Select the logical group to apply the throttle definitions to from the **File** menu.
2. Select the **Throttles** tab.

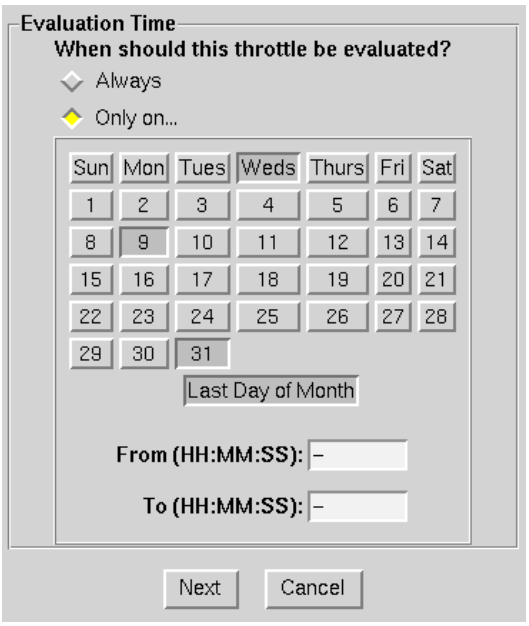
The Throttles Editor



How to Use Throttle Builder

1. If you are building throttles for the first time, select the **Throttle Builder** button.
2. Define when throttles are to be evaluated. The default is **Always**. To change the default select **Only On** and use the calendar to select days of week or days of month on which you want the throttle to be active as shown in the following figure.

Setting the Evaluation Time through the Throttle Builder



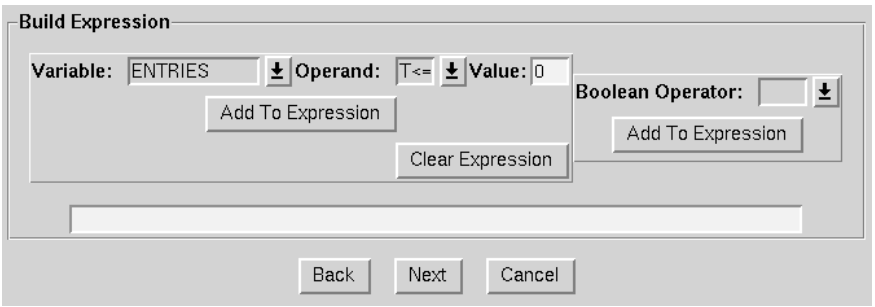
3. Specify a **From** and **To** time of day for the throttle to be active using the HH:MM:SS format by placing the cursor in the field and entering the appropriate hour (HH), minute(MM) and second (SS) values. Accepting the "-" symbol default makes the throttle active continuously during the days that are selected in step 2.

How to Build Expressions

▶ **To build expressions:**

1. Select **Next** to access the **Build Expressions** screen.

Throttle Expressions



2. Choose a **Variable** from the drop-down menu. This is the element to be measured or monitored by the throttle.

3. Choose an **Operand** from the drop-down menu. This element defines the relationship between the **Variable** and the **Value**.
4. In the **Value** field, enter an appropriate value.
5. Select the **Add to Expression** button.

NOTE If you make mistake in building the throttle expression, use the **Clear Expression** button to reset the editor.

▶ **To create a compound expression:**

1. Select the applicable **Boolean Operator** (and/or) from the drop-down menu.
2. Select the **Add to Expression** button.
3. Define second half of the expression by defining a **Variable**, **Operand** and **Value**.
4. Repeat steps 1-3 until reaching the desired result or the maximum number (16) of clauses.

How to Build Actions

▶ **To build actions:**

1. Select **Next** to access the **Build Actions** screen.

Building Throttle Actions with the Throttle Builder

2. Choose an **Action** from the drop-down menu.
3. Choose a **Variable** from the drop-down menu.
4. Enter an **Argument** in the field.
5. Select the **Add to Action List** button.

NOTE Each throttle can have up to 16 defined actions.

6. Repeat these steps until all **Actions** are defined for this specific throttle.
7. Select the **Done** button. The initial Throttle screen is displayed with the defined throttle visible in the **Throttle Editor** window.

When you are ready for the new throttle definitions to take effect, perform the following steps:

8. Terminate user applications and unmount any file systems accessing the dtc devices in the affected logical group.
9. Terminate the PMD for the logical group: `killpmds -g <group#>`.
10. Type the following:
    `dtcstop -g <group#>`
    `dtcstart -g <group#>`
11. Start the PMD with the `launchpmds -g <group#>`.
12. Start any user applications or mount file systems using devices in the logical group.

Throttle Examples

This section includes throttle examples to illustrate the value of throttles in a Softek Replicator production environment. These throttle examples are provided for illustration purposes only. When defining throttles, you must balance all critical business factors that may affect applications and apply appropriate throttle conditional tests and actions to cover all situations that might arise. Carefully research and test throttles before implementing them in a production environment.

The throttle shown in the following figure is arranged to restrict maximum network utilization to 300 KBps during the standard business day, but lift the restriction outside those hours.

Throttle to Regulate Maximum Network Bandwidth during Peak Business Hours

```
THROTTLE: MO,TU,WE,TH,FR 08:00:00 17:00:00 \
NETMAXKBPS != 300
ACTIONLIST:
ACTION: set NETMAXKBPS 300
ENDACTIONLIST:
THROTTLE: MO,TU,WE,TH,FR 17:01:00 23:59:59 \
NETMAXKBPS != 1000000
ACTIONLIST:
ACTION: set NETMAXKBPS 1000000
ENDACTIONLIST:
THROTTLE: MO,TU,WE,TH,FR 00:00:00 07:59:59 \
NETMAXKBPS != 1000000
ACTIONLIST:
ACTION: set NETMAXKBPS 1000000
ENDACTIONLIST:
THROTTLE: SA,SU - - NETMAXKBPS != 1000000
ACTIONLIST:
ACTION: set NETMAXKBPS 1000000
ENDACTIONLIST:
```

The figure: *Throttle to Restrict Network Usage during Peak Business Hours* restricts maximum network utilization to 300 KBps during the standard business day, but lifts the restriction outside those hours.

Throttle to Restrict Network Usage during Peak Business Hours

```
#
THROTTLE:  Fr,Tu,Th,Mo,We 08:00:00 17:00:00 \
   NETMAXKBPS != 300
ACTIONLIST
   ACTION: set NETMAXKBPS 300
ENDACTIONLIST

THROTTLE:  Fr,Tu,Th,Mo,We 17:01:00 23:59:59 \
   NETMAXKBPS != 1000000
ACTIONLIST
   ACTION: set NETMAXKBPS 1000000
ENDACTIONLIST

THROTTLE:  Fr,Tu,Th,Mo,We 00:00:00 07:59:59 \
   NETMAXKBPS != 1000000
ACTIONLIST
   ACTION: set NETMAXKBPS 1000000
ENDACTIONLIST

THROTTLE:  Sa,Su - - NETMAXKBPS != 1000000
ACTIONLIST
   ACTION: set NETMAXKBPS 1000000
ENDACTIONLIST
#
```

Changing Tunable Parameters

Tunable parameter values are saved in the pstore. These values can be set on the fly using `dtcconfigtool` or with the `dtcset` command.

Using dtcset

Note that you must run `killpmds` before and `launchpmds` after changing the following tunable parameters:

- CHUNKSIZE
- COMPRESSION
- NETMAXKBPS

Note that you must run `dtcstop` before and `dtcstart` after changing the following tunable parameters:

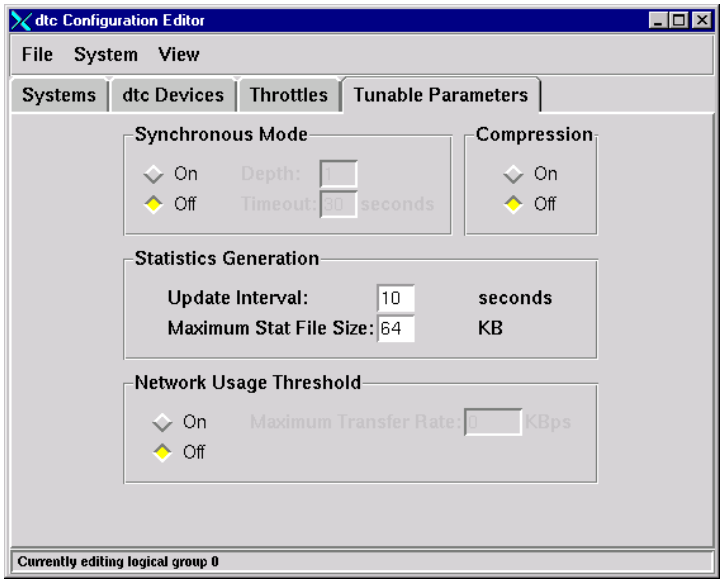
- SYNCMODE
- SYNCMODEDEPTH
- SYNCMODETIMEOUT

Type `dtcset -g <group#> <tunable=value >`.

Using dtcconfigtool

1. Run `dtcconfigtool`.
2. Select the logical group to update from the **File** menu.
3. Select the **Tunable Parameters** tab as shown in the following figure.

Tunable Parameters Tab in dtcconfigtool



4. Edit definitions.
5. From the **File** menu, **Save Changes** and **Exit**.
6. Type `killpmds -g <group#>`.
7. Unmount any file systems and stop any applications accessing the dtc devices in the logical group.
8. Run `dtcstop -g <group#>`, followed by `dtcstart -g <group#>`.
9. Restart the PMDs with `launchpmds`.
10. Remount the file system and restart applications.

Adding/Modifying Devices

▶ **To add or modify dtc devices while Softek Replicator is running:**

1. Run `dtcconfigtool`.
2. Select the logical group you want to add, or modify dtc devices from the **File** menu.
3. Select the **dtc Devices** tab.
4. Choose the dtc device to be modified from the device list on the left side of the screen. The selected device is highlighted and the device information is displayed in the fields to the right.
5. Select a new device from the drop-down menus or enter a new value.
6. To add a new dtc device, select the **Create new device** button and enter the local data device and mirror device definitions.
7. From the **File** menu, select **Save Changes**.
8. If the local data device is already mounted, a dialog box will appear to confirm the unmount for the device. If it's ok to unmount, please select **Yes**.
9. If the local data device is registered for auto-mount in `FSTAB` file, a dialog box will appear to confirm the change of mount information. If it's ok, please select **Yes**.

NOTE

FSTAB File

The `FSTAB` file mentioned above is `/etc/vfstab` for Solaris, `/etc/fstab` for HP-UX and Linux, and `/etc/filesystems` for AIX.

Steps 8 and 9

Softek Replicator does not perform Steps **8** and **9**, above, if one of the following is true:

- The selected logical group is started
- The data device is already registered to the selected logical group

Further, Step **9** is only performed under the following conditions:

- AIX

When the data device of the specified logical group exists; the mount stanza of that record is defined to `[true] [automatic]`; and the dev stanza or the log stanza agrees with the device name.

- HP-UX

When the data device of the specified logical group exists, and the file system type of that record is other than `[swap] [swapfs] [dump] [ignore] [nfs]`.

- Linux

When the data device of the specified logical group exists, and the file system type of that record is other than `[swap] [proc] [tmpfs] [devpts] [dump] [ignore] [auto] [nfs]`.

- Solaris

When the data device of the specified logical group exists, and the automount is specified to `yes`.

During the update process, any data that is changed is first converted into a comment line and (Replicator-specific) character string, and the date of the update is added to the end of the record.

10. Copy the configuration files over to the secondary system and rename them.
11. Type `killpmds -g <group#>`.
12. Unmount any file systems and stop any application from accessing the dtc devices in the logical group.
13. Run `dtcstop -g <group#>`.
14. Run `dtcstart -g <group#>`.

NOTE Remember that the configuration files are only processed by the `dtcstart` command. Changes to the configuration do not take effect until the logical group is stopped and restarted.

15. Mount file systems or start applications.
16. Run `launchpmds -g <group#>`.

17. If new devices have been added to the logical group, run
    `launchrefresh -g <group#> -f`.

▶ To remove dtc Devices:

1. Run `dtcconfigtool` on the primary system.
2. Select the logical group for modification from the **File** menu.
3. Select the **dtc Devices** tab.
4. Choose the dtc device to be deleted from the device list window.
5. Select **Delete Device**.
6. From the **File** menu, select **Save Changes**.
7. Copy the update configuration file (`.cfg`) from the primary to the secondary system.
8. Type `killpmds -g <group#>`.
9. Unmount any file systems and stop any application from accessing the dtc devices in the logical group.
10. Type `dtcstop -g <group#>`.
11. Type `dtcstart -g <group#>`.

NOTE The changes to the logical group do not take effect until the group has been stopped and restarted - when `dtcstart` creates a new `.cur` file.

12. Mount file systems or start applications.
13. Run `launchpmds -g <group#>`.

Deleting Logical Groups

▶ To delete a logical group:

1. Type `killpmds -g <group#>` to stop the logical group you want to delete.
2. Unmount the file systems that are mounted on any dtc devices belonging to the logical group.
3. Type `dtcstop -g <group#>`.
4. Remove the appropriate `p###.cfg` configuration file (where ### is the number of the logical group to be deleted). For example:

   On AIX: `rm /etc/dtc/lib/p01.cfg`

   On HP-UX, Linux, and Solaris: `rm /etc/opt/SFTKdtc/p001.cfg`

The next time you reboot the system, the blocks used by the deleted logical group will be available.

Relocating the pstore

To relocate the pstore volume, run `dtcconfigtool` and select a new pstore volume from the drop-down menu for each logical group, for which the pstore is being modified.

1. Shut down any processes and unmount any file systems accessing the dtc devices in the affected logical groups.
2. Type `killpmds -g <group#>` for each group affected by the pstore modification.
3. Type `dtcstop -g< group#>` for each group affected by the pstore modification.
4. Start `dtcconfigtool`.
5. Select the **Systems** tab, and choose a new pstore volume from the drop down menu.
6. From the **File** menu, select **Save Changes**.
7. Type `dtcstart -g <group#>` for each modified group.
8. Restart the groups with a `launchpmds -g <group#>`.
9. Remount files systems and restart applications.

Modifying the BAB

To resize the BAB, you must reload the dtc driver. Before doing this, shut down any processes accessing the dtc devices, unmount any file systems residing on the dtc devices, and stop all logical groups using the `dtcstop` command. All Softek Replicator daemons are temporarily stopped while this reload takes place.

Using dtcconfigtool

1. Run `dtcconfigtool`.
2. From the **Systems** pulldown menu, select **BAB**.
3. Use arrows to select designated amount of memory or enter a new amount.
4. Select **Ok**.
5. If the newly requested memory amount cannot be allocated, reboot the system.
6. Restart applications.

Manually

1. Unmount filesystems and kill applications accessing dtc devices.
2. Type `killdtcmaster`.

NOTE **For Linux only**
Unload the dtc driver by typing:
`/sbin/rmmod/sftkdtc`

3. For HP-UX and Linux only: Resize the BAB with `dtcinit -b <memory in MB>`. In HP-UX, this command executes a kernel rebuild. In Linux, this command modifies `/etc/modules.conf` so that after the module is reloaded, this system reads new values and initializes the BAB.

NOTE The only way to specify a BAB size outside the standard selections in the Configuration Tool dialog is to enter the `dtcinit -b` command.

4. Run `dtcinfo` to verify that memory was acquired.
5. Type a `launchdtcmaster` command.
6. Remount filesystems and restart applications.

   If the BAB overflows during a refresh operation, you will see the following message:

   ```
   RFDOFLOW BAB exhausted during a refresh operation <processname>. Run launchrefresh for the group.
   ```

   Type the `launchrefresh` command to restart the logical group.

If the BAB overflows during a smart refresh, Softek Replicator is placed in TRACKING mode. You must type the `launchrefresh` command to move the product back to NORMAL mode. If the BAB overflows frequently, consider resizing the BAB and adding network resources.

Resizing the Journal File Directory

1. From the primary system, type `killpmds` and then `killdtcmaster`.
2. On the secondary system, type `killdtcmaster`.
3. Remove the journal file(s); for example:
   `rm /journals/j001.002.c`
4. Mount a larger journal file directory.
5. From the primary and secondary systems, type `launchdtcmaster`.
6. From the primary system, type `launchpmds`.

Softek Replicator automatically launches a smart refresh to synchronize the primary and secondary systems, applies the new journal file to the mirror device, updates journals during the smart refresh, and then goes to normal mode.

Checkpointing Softek Replicator

A *checkpoint* is a frozen snapshot of a data set at a specific point in time. A checkpoint allows you to take the secondary mirror off-line from Softek Replicator and make it available for read-only access by other applications. *Checkpointing* ensures that the data on the mirror device is in a known, usable state and that no updates are lost while the mirror is not available to Softek Replicator. While the checkpoint is in effect, all transactions targeted at the mirror devices are diverted to the journal on the secondary system.

NOTE You can type the `dtccheckpoint` command on either the primary or secondary systems with the same result.

Mount file systems on "checkpointed" mirror devices in *read-only* mode. Likewise, if you run `fsck` on the mirror devices, it must be a report-only, non-modifying `fsck`.

Examples of Checkpoints

For AIX:

```
aixserv# fsck -n /dev/lv12
aixserv# mount -o ro /dev/lv12 /mirror_mount
```

For HP-UX:

```
hpserv# fsck -n /dev/vg00/lv12
hpserv# mount -o ro /dev/vg00/lv12 /mirror_mount
```

For Linux:

```
linuxserv#  /sbin/fsck /dev/hda5
linuxserv#  mount -o ro /dev/hda5 /mirror_mount
```

For Solaris:

```
sunserv# fsck -n /dev/rdsk/cotod0s#
sunserv# mount -o ro /dev/rdsk/c0t0d0s# /mirror_mount
```

When the checkpoint has been turned off, the RMD daemons reacquire their exclusive open lock on the mirror devices and immediately apply the accumulated transactions from the secondary journals, bringing the mirror devices up to date with the primary local data devices.

Checkpoint Processing

The `dtccheckpoint` command is used to activate the checkpoint feature for Softek Replicator. This command can be typed on either primary or secondary systems with the same result. Before entering the `dtccheckpoint` command with the appropriate options, applications on the primary must be quit so that all data can be flushed to disk. You can do this either manually or via a checkpoint shell script that the checkpoint command calls automatically. For more information on preparing both systems for a checkpoint operation, see *Checkpoint Shell Scripts* on page 71.

The following process takes place when the checkpoint is turned ON:

1. The `dtccheckpoint -on` command is typed either from the primary or secondary machine. If you have defined checkpoint scripts, the `/etc/opt/SFTKdtc/cp_pre_on_p###.sh` script is run at this time. If this script file is not defined, you must manually quit the applications on the primary and force all pending updates to the disk.

NOTE Before entering the `dtccheckpoint -on` command on the primary system, you must stop the database, and unmount and remount all applications accessing dtc devices in the logical groups being placed into checkpoint. This ensures that application data is flushed to disk and that the mirror device is in a known state.

2. Once all data has been flushed to disk and to the BAB on the primary, a flag is set in the BAB. This flag alerts the RMD to start "journaling" all subsequent entries received by the secondary.

3. Operations that were interrupted on the primary system in order to flush the data should be brought back into a normal operating state. You can create an `/etc/opt/SFTKdtc/cp_post_on_p###.sh` script to automate this process.

4. On the secondary system, updates are being "journaled" and the RMD has released its exclusive lock on the mirror devices, which can now be mounted and accessed in a read-only fashion by applications. You can create an `/etc/opt/SFTKdtc/cp_post_on_s###.sh` script to automate mounting or accessing these mirror devices on the secondary system. This is a useful way to perform a backup of the mirror devices without interrupting the primary system for a prolonged period.

The following process takes place when the checkpoint is turned OFF:

1. Before entering the `dtccheckpoint -off` command, file systems or applications accessing the mirror devices on the secondary system must be stopped and the file systems unmounted. (If the mirror device(s) is mounted, checkpoint is NOT turned OFF). The `/etc/opt/SFTKdtc/cp_pre_off_s###.sh` script can be used to automate this process.

2. The mirror devices are reacquired with an exclusive lock by the RMDs and the journal entries that have been accumulating in the secondary journal file directory are now applied to the mirror devices as appropriate.

NOTE A mirror device is in an incoherent but recoverable state while these journal entries are being applied.

Checkpoint Shell Scripts

NOTE For AIX only, replace `/etc/opt/SFTKdtc` with `/etc/dtc/lib` in the shell script.
For example: `/etc/dtc/lib/cp_pre_on_p###.sh`

To automate the process that ensures data set consistency at the time of checkpointing, you may choose to implement one or more optional checkpoint shell scripts.

For a data set to be usable on the secondary system in checkpoint mode, the data must be complete and in a consistent state. This often means that applications such as databases must be momentarily shut down to bring the data set to a known, good state. The checkpoint facility does this automatically for file systems on the primary system by flushing any uncommitted blocks in the buffer cache to disk before entering into checkpoint mode.

Once the `dtccheckpoint -on` command adds the `checkpoint -on` token to the BAB, then applications on the primary system can resume data updating activities.

Primary System Scripts

The process of preparing the data set for checkpointing, placing the token in the BAB and resuming normal application operations on the primary system should take only a few seconds (or a few minutes for a highly complex database environment). This process can be automated using optional shell scripts for the primary system.

The location and names of the following shell scripts on the primary system are predefined and cannot be changed:

- `/etc/opt/SFTKdtc/cp_pre_on_p###.sh` is run on the primary system (denoted by the p### naming convention) before activating the checkpoint to prepare the specified logical group. It is automatically called when the `dtccheckpoint -on` command is typed.
- `/etc/opt/SFTKdtc/cp_post_on_p###.sh` is run on the primary system after the `dtccheckpoint -on` command has placed a marker in the BAB. This marker notifies the RMD of the checkpoint request. At this point, all future updates to the RMD are sent to the journal file area on the secondary system, until the `dtccheckpoint -off` token is received. Use this script to bring applications back online on the primary system.

NOTE The `#` symbol corresponds to the logical group number to which this shell script applies. For instance, for logical group 12, the `cp_pre_on_p012.sh` shell script would be run before creating the checkpoint.

The figure: *Script Sample of /etc/opt/SFTKdtc/cp_pre_on_p012.sh for Primary System* and the figure: *Script Sample of /etc/opt/SFTKdtc/cp_post_on_p012.sh for Primary System* show sample primary system checkpoint shell scripts.

Script Sample of /etc/opt/SFTKdtc/cp_pre_on_p012.sh for Primary System

```
#!/bin/sh
# ***** /etc/opt/SFTKdtc/cp_pre_on_p012.sh *****
echo ---------- `date` ---------- >> /usr/tmp/
xpoint.log
echo starting cp_pre_on_p012.sh >> /usr/tmp/
xpoint.log
/oracle/script/shutdown_database.sh
echo database has been shut down >> /usr/tmp/
xpoint.log
echo " " >> /usr/tmp/xpoint.log
exit 0
```

Script Sample of /etc/opt/SFTKdtc/cp_post_on_p012.sh for Primary System

```
#!/bin/sh
# ***** /etc/opt/SFTKdtc/cp_post_on_p012.sh *****
echo ---------- 'date' ---------- >> /usr/tmp/
xpoint.log
echo starting cp_post_on_p012.sh >> /usr/tmp/
xpoint.log
/oracle/script/startup_database.sh
echo database has been restarted >> /usr/tmp/
xpoint.log
echo " " >> /usr/tmp/xpoint.log
exit 0
```

NOTE **Error Messages:**
These scripts must be executable (have an "x" permission attribute) and return an exit code of "0" to be properly executed by the `dtccheckpoint` command. An error message is issued if there is any problem with the execution of these shell scripts.

Secondary System Scripts

To automate the process of "checkpointing" information as it becomes available on the secondary system, you may choose to implement one or both optional shell scripts that are run on the secondary system:

- `/etc/opt/SFTKdtc/cp_post_on_s###.sh` is run immediately after the `checkpoint -on` token is received by the RMD. At this point, all updates received by the RMD are directed to the secondary journal area. You can use this script to launch a tape backup of the mirror, to mount file systems (in *read-only mode*) on the mirror devices, or to start a database in read-only mode for report generation or decision support applications.
- `/etc/opt/SFTKdtc/cp_pre_off_s###.sh` is run immediately before the RMD regains exclusive control of the mirror devices after the `dtccheckpoint -off` command is issued. You can use this script to unmount the read-only file systems on the mirror devices, to stop database applications running for report generation or decision support applications, or to terminate applications on the secondary so that the RMDs can regain control of the mirror devices.

The figure: *Script Sample of /etc/opt/SFTKdtc/cp_post_on_s012.sh for Secondary System* and the figure: *Script Sample of /etc/opt/SFTKdtc/cp_pre_off_s012.sh for Secondary System* show an example set of secondary scripts that fit the context of the previous primary script examples. In the following example, a level 0 tape backup is performed on the checkpointed data set. Note that this procedure has a minimal impact on the primary system (momentary shutdown for synchronization).

Script Sample of /etc/opt/SFTKdtc/cp_post_on_s012.sh for Secondary System

```
#!/bin/sh
# ***** /etc/opt/SFTKdtc/cp_post_on_s012.sh
*****
echo ---------- 'date' ---------- >> /usr/tmp/
xpoint.log
echo starting cp_post_on_s012.sh >> /usr/tmp/
xpoint.log
echo starting full level-0 cold back-up of data-
base \
>> /usr/tmp/xpoint.log
/usr/local/bin/db_backup.sh
echo database backup completed, exiting check-
point \
>> /usr/tmp/xpoint.log
echo " " >> /usr/tmp/xpoint.log
/opt/SFTKdtc/bin/dtccheckpoint -g 12 -off
exit 0
```

Script Sample of /etc/opt/SFTKdtc/cp_pre_off_s012.sh for Secondary System

```
#!/bin/sh
e# ***** /etc/opt/SFTKdtc/cp_pre_off_s012.sh
*****
cho ---------- 'date' ---------- >> /usr/tmp/
xpoint.log
echo starting cp_pre_off_s012.sh >> /usr/tmp/
xpoint.log
if [ -f /tmp/db_backup_PID ]; then
echo database back-up error, killing it \
>> /usr/tmp/xpoint.log
/bin/kill -9 </tmp/db_backup_PID
fi
echo " " >> /usr/tmp/xpoint.log
exit 0
```

Checkpoint Scripts for File Systems

The figure: *File System Checkpoint Script - /etc/opt/SFTKdtc/cp_post_on_s001.sh* and the figure: *File System Checkpoint Script - /etc/opt/SFTKdtc/cp_pre_off_s001.sh* show checkpointing scripts written to make a file system available for examination, reading, copying or for other applications. These scripts are not mandatory—you can do these steps manually.

Primary Checkpoint Scripts

No scripts are required on the primary. The checkpoint command issues a UNIX `sync` command to flush uncommitted updates from the buffer cache to disk prior to inserting the checkpoint token into the BAB. If an active application, such as a database, requires additional quieting steps, then you can write primary scripts to automate the process.

Secondary Checkpoint Scripts

The following figures show examples of the scripts that can be defined on the secondary to automate the procedures for checkpointing file systems.

File System Checkpoint Script - /etc/opt/SFTKdtc/cp_post_on_s001.sh

```
#!/bin/sh
# ***** /etc/opt/SFTKdtc/cp_post_on_s001.sh *****
echo ---------- `date` ---------- >> /usr/tmp/
xpoint.log
echo starting cp_post_on_s001.sh >> /usr/tmp/
xpoint.log
#
# -- mount the mirror devices read-only
mount -o ro /dev/dsk/c1t4d0s5 /mnt_rmt
echo /mnt_rmt file system mounted read-only >> \
/usr/tmp/xpoint.log
echo " " >> /usr/tmp/xpoint.log
#
# -- this is a good spot to start any application
that
# -- needs to work with the checkpointed file sys-
tem data
# -- on the secondary or to notify users that this
data
# -- is now available
exit 0
```

File System Checkpoint Script - /etc/opt/SFTKdtc/cp_pre_off_s001.sh

```
#!/bin/sh
# ***** /etc/opt/SFTKdtc/cp_pre_off_s001.sh *****
echo ---------- `date` ---------- >> /usr/tmp/
xpoint.log
echo starting cp_pre_off_s001.sh >> /usr/tmp/
xpoint.log
#
# -- this is a good spot to stop any application
using the
# -- checkpointed file system and to shoo users away
from it
# -- (with force, if necessary).
#
umount /mnt_rmt
echo /mnt_rmt file system un-mounted >> /usr/tmp/
xpoint.log
echo " " >> /usr/tmp/xpoint.log
exit 0
```

In addition to the four shell scripts described, you can create an `etc/opt/SFTKdtc/cp_reboot_s###.sh` script to automate actions when a reboot occurs while checkpoint is on. This script can simply enter a `dtccheckpoint -off` command or verify that a backup completed before doing a reboot. If it detects that the operation did not complete, it may relaunch the application.

NOTE If data on the mirror device(s) is altered by applications accessing the devices while the checkpoint is on, then the mirror is broken and manual intervention is required to synchronize the systems. The actual process may require a full refresh, a backfresh or a backup/restore operation depending on the how the data is prioritized on each system.

Tips for Checkpoint Scripts

- Scripts intended to run on the primary have a `p` before the logical group number and scripts intended to run on the secondary have an **s** before the logical group number.
- Script files must have the **x** executable attribute set.
- All scripts must end with `exit 0` or the `dtccheckpoint` command fails and the operation is aborted.
- Define only the checkpoint scripts that you need. You need not define all four scripts.

Considerations and Limitations

- Checkpoint operations are not immediate. The checkpoint -on token is queued in the BAB on the primary with data update transactions. Only after the RMD receives this token does the checkpoint go into effect. When the amount of data in the BAB awaiting transfer is large, there may be a noticeable delay before the checkpoint takes effect.
- If a failure occurs on the primary system or an explicit `dtcstop` is entered to a logical group with a pending checkpoint token in the BAB, the token is lost when the logical group is rebooted/restarted and the checkpoint operation is cancelled.
- After the `dtccheckpoint` command has been entered, but before the `dtccheckpoint` token is transmitted to the secondary system, there is no graceful way to cancel the requested checkpoint operation. The operation is irrevocable at this point without killing the PMD/RMD daemons, deactivating the logical group by typing an `dtcstop` command, clearing the BAB with the `dtcoverride` command, or re-initializing the primary system. Any one of these actions may require refresh operations to bring the systems back into synchronization.
- Activating or deactivating checkpointing can only occur if the logical group is in NORMAL MODE. The logical group must remain in NORMAL MODE until the checkpoint token is received by the RMD on the secondary machine at which time it is acceptable for the system to go to a new operating mode.
- If the BAB fills up and forces the logical group into TRACKING or REFRESH MODE while the checkpoint token is queued for transfer, the operation is cancelled. Note that, if defined, the `cp_pre_on_p###.sh` script has been run at this point. Therefore, you must manually execute the `cp_post_on_p###.sh` script on the primary to place applications in a pre-checkpoint state.

- Softek Replicator examines the secondary journal area for the logical group to determine whether the secondary system is in a checkpoint state. The presence of any files with a `.p` file extension indicates that checkpoint is active.
- Should your application modify the mirror devices while in checkpoint mode, a full refresh is required. There is no way around this.

Mounting Mirror Devices on HP-UX and AIX

On HP-UX and AIX, mirror devices for a particular logical group are allowed to be mounted with read/write permission even though the logical group is in Normal mode; whereas, Softek Replicator for Solaris prevents the same mirror devices from being mounted.

Should this mount happen and some data is written to the mounted file system, it will cause the data to be inconsistent between the primary and secondary devices. Any mirror device that is defined by the logical group in Normal mode must not be mounted as read/write.

Rebooting the Primary System on HP-UX

▶ **To reboot the primary system, enter:**

```
shutdown -r
```

Do not use the `reboot` command to restart or reboot the primary system.

NOTE **Stopping Applications That Use the dtc Device**

Applications (or processes) that use the dtc device must be stopped, either manually or automatically via the `rc` script, before the primary system is rebooted. To auto-stop a specific application or process via the `rc` script, set the application's stopping procedures before `/etc/rc0.d/k800SFTKdtc-scan`.

4

Recovering Data

Scenario 1: Secondary System/Network Failure 81

Scenario 2: Primary System Failure 82

Scenario 3: Failure to Transition Out of Tracking Mode 83

Scenario 4: Failing Over to the Secondary System, Restoring the Primary System 83

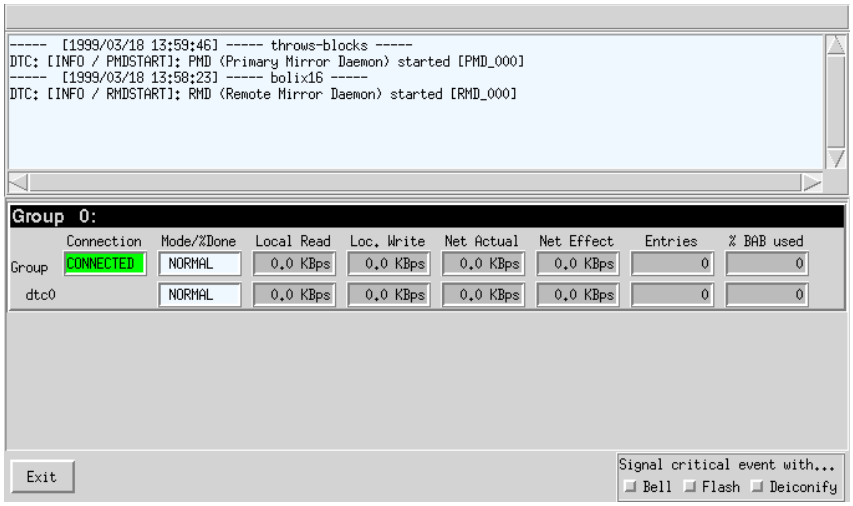
Scenario 5: Device Failure on the Primary or Secondary System 84

This chapter describes how to recover your data in case:

- your primary or secondary system fails;
- a device on either system fails;
- the network between your systems is unavailable;
- you need to restore the primary system after a changeover to the secondary system.

Scenario 1: Secondary System/Network Failure

If the secondary system or network has failed, the **Connection** field in `dtcmonitortool` turns red and indicates **Accumulate** mode.



If the secondary system or network becomes unavailable for a prolonged period of time, do one of the following:

- If the **Mode** field indicates that the product is still in NORMAL mode:

  a. When the secondary system comes back into service, use the `launchpmds` command from the primary system to start the PMD and RMD.

  b. Softek Replicator will continue to operate in NORMAL mode and will begin mirroring the data again.

- If the **Mode** field indicates that the product is in TRACKING mode:

  - When the secondary system comes back into service, use the `launchrefresh` command from the primary system to perform a smart refresh to resynchronize the systems.

- If the secondary system must be replaced, create a new secondary system and network connection to receive the mirrored data as follows:

  a. Install Softek Replicator on the new secondary system according to the instructions in the *Installation Guide*.

  b. Change the configuration of each defined logical group with `dtcconfigtool` (on the primary system) so that they refer to the new secondary system.

Follow the process outlined in *Chapter 2: Configuration* as it pertains to setting up the new secondary system. Remember to copy the updated configuration file to the new secondary system.

c. Follow either synchronization procedure described in *Chapter 2: Configuration* to create the initial mirror from the primary to the new secondary system.

Scenario 2: Primary System Failure

If a failure occurs on the primary system that places it out of operation for hours, days, or longer, data can be accessed from the secondary system.

▶ **To allow data to be accessed from the secondary system:**

1. If possible, stop the PMDs by entering a `killpmds` command on the primary system.
2. Commit any buffered updates to the mirror devices using a `dtcrmdreco -g <group#>` command from the secondary system.

NOTE You must run the `dtcrmdreco` command, because it ensures data integrity on the mirror devices and relinquishes the exclusive hold that Softek Replicator has on these devices.

3. If the mirror devices contain a file system, `fsck` those devices and `mount <filesystems>`.
4. Start application(s) on the secondary system.
5. See *Scenario 4: Failing Over to the Secondary System, Restoring the Primary System* on page 83 for instructions on restoring the primary system to primary status.

Switching to the Secondary System with an RDBMS

If you are using Softek Replicator to mirror data from a database (as described in *Configuring Softek Replicator for Relational Database Management Systems (RDBMS)* on page 53) follow these steps to change over to the secondary system.

1. On the secondary system, type a `killrmds` command to ensure that no Softek Replicator process has mirror devices locked.
2. Type `dtcrmdreco -a` to ensure that all uncommitted Softek Replicator data actually gets written to the mirror devices.
3. Create the necessary symbolic links to now point to the mirror devices. For example:

```
$ mkdir /dev/devlinks
$ ln -s /dev/rdsk/c1t1d0s1 /dev/devlinks/tblspA
$ ln -s /dev/rdsk/c1t2d0s1 /dev/devlinks/tblspB
$ ln -s /dev/rdsk/c1t3d0s1 /dev/devlinks/tblspC
```

Change the ownership of the mirror devices to the RDBMS.

`/dev/rdsk/c1t1d0s1` is actually a symbolic link itself. Use the command `ls -la /dev/rdsk` to see where the device instance actually is, `cd` to that directory and change the device ownership appropriately. Now you can bring up the database on the secondary system. The database performs cleanup and rollback of any uncommitted partial transactions that it finds as part of its standard recovery process.

Scenario 3: Failure to Transition Out of Tracking Mode

During normal operation, if the BAB overflows for a group on the primary system, the following message is displayed in the `dtcmonitortool` window or in the dtcerror.log:

```
BABOFLOW BAB exhausted during normal operation <processname>.
```

The group is temporarily placed in TRACKING mode. Then, the PMDs drain the BAB entries, the group transitions to a smart refresh, and transitions back into NORMAL mode after the refresh is complete. No user intervention is required.

In the case of a double BAB overflow - a second BAB overflow occurs while the group is still in TRACKING or REFRESH mode. In this case, the following message is displayed:

```
RFDOFLOW BAB exhausted during a refresh operation <processname>. Run
launchrefresh for the group.
```

With a double BAB overflow, the group is placed in TRACKING mode, in which it remains until you type:

```
launchrefresh -g <group#>
```

The group goes to a smart refresh and then back to NORMAL mode. After all journal files are applied to the mirror devices, the primary and secondary systems are in a coherent state.

Note that if you experience double BAB overflows, you may need to increase the size of your BAB. For more information, see *Modifying the BAB* on page 68.

Be aware that these messages may have scrolled off the `dtcmonitortool` screen.

Scenario 4: Failing Over to the Secondary System, Restoring the Primary System

Assume that you had a disaster on the primary system and want to relocate operations to a secondary system using the mirrored data. Follow these steps:

1. Issue `dtcrmdreco -g <group#>` on the secondary system.
2. If necessary, mount mirror devices. Then start applications on the secondary system.

Now your primary system is back in order, and you want to move operations back to the primary system. Synchronize the primary system with the more recent data on the secondary data devices as follows:

1. Halt applications and unmount the devices on the secondary system in preparation for restoring the primary.
2. Enter `dtcrmdreco -g <group#> -d` on the secondary system.
3. On the primary system, type:
   `$ /opt/SFTKdtc/bin/launchbackfresh -g <group#>`

When the backfreshcompletes, Softek Replicator shifts into NORMAL mode, the PMDs and RMDs are connected, and you can resume mirroring data to the secondary system.

NOTE Backfresh is an operation in maintenance mode only. Do not access or update local or mirror data devices until the backfresh is complete.

Alternately, you can bring the primary system back into synchronization with the more recent data on the secondary system by switching the primary and secondary systems roles and performing a refresh operation from the original secondary system back to the original primary one.

One advantage of this approach is that applications can continue to update the data on the original secondary system during the refresh operation. For more information, refer to *Complex Configurations* on page 119.

Scenario 5: Device Failure on the Primary or Secondary System

You can detect a device failure on one of your systems using a tool such as Volume Manager. The device appears grayed if it has failed. You may also see a Softek Replicator message to this effect in dtcerror.log.

In this case, take the following steps:

1. If you are using file systems, unmount the dtc devices in the affected logical group.
2. Type `killmpds -g <group#>`.
3. On the primary system, stop the logical groups affected by the device failure using the `dtcstop -g <group#>` command.
4. Add or replace the device on the system.
5. Using `dtcconfigtool`, modify the affected logical group definitions so that they refer to the new device. Remember to copy the new .cfg file to the secondary system.
6. Restart the logical groups with the `dtcstart -g <group#>` command.
7. If the device failed on the primary system, type:
   `launchbackfresh -g <group#>`

   If the device failed on the secondary system, type:

   `launchrefresh -f -g <group#>`

5

Monitoring Tools

dtcperftool 87

dtcmonitortool 90

dtcinfo 93

Softek Replicator provides three utilities that let you monitor the system as it is running. These tools allow you to observe the effect that current tunable parameters and throttles have on the system. Monitoring can also alert you to problems as they occur in initial configurations or due to the expansion of existing Softek Replicator environments.

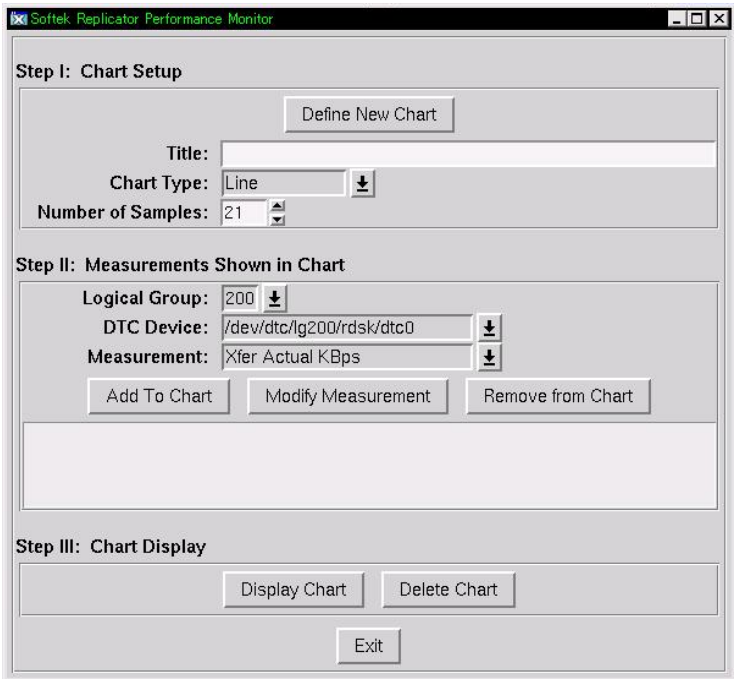
dtcperftool

The `dtcperftool` is used to display charts of various Softek Replicator performance metrics. You can view multiple charts at one time, and modify, delete or print them. `dtcperftool` lets you observe Softek Replicator performance and trends over time, and create printouts.

Moreover, you can run `dtcperftool` on the primary system, secondary system, or both. Thus, you can monitor data being sent and received. If primary and secondary configuration files exist on the system when `dtcperftool` is run, use the optional `-p` or `-s` arguments to determine whether the primary or secondary configuration files are queried. The only difference in using the tool on the systems is what performance metrics you can observe. To start the performance monitoring tool, enter `dtcperftool` from the command line.

The `dtcperftool` screen is divided into three sections, or steps, corresponding to the steps you follow to generate a performance monitoring chart.

Softek Replicator Performance Monitor Window



Step I: Chart Setup

- If you are creating a new chart, select **Define New Chart** to clear all the fields.
- Specify the following:
  - **Title**: Title for the chart you are creating—up to 60 characters.
  - **Chart Type**: How you want the data displayed. Select one of the following chart types from the drop-down menu:
    - Line Chart
    - Bar Chart
    - Stacked Bar Chart
  - **Number of Samples**: The number of samples to be shown in a chart.

    The PMD writes performance statistics to an ASCII file. By default, this file is updated every 10 seconds. Thus a 21-sample analysis reflects 210 seconds of operation.

Step II: Measurements Shown in Chart

- Specify the following items to show in the chart.

  Select a **Measurement** and click **Add To Chart**. Select as many measurements as you wish. The combination of logical group, dtc device, and measurement type is displayed in the box beneath the **Add To Chart** button.
  - **Logical Group** — The logical group(s) you want monitored. Select one of the currently defined logical groups from the drop-down menu.
  - **DTC Device** — The dtc device(s) you want monitored. Select a dtc devices defined for the logical group you have selected
  - **Measurement** — The data item(s) you want displayed. Choose as many types of data as you want from the **Measurement** drop-down menu. The measurements you can select for the primary system are:
    - **Actual KBps** — The actual number of KBytes of data sent over the network.
    - **Effective KBps** — If you have compression activated with the COMPRESSION tunable parameter, the effective KBps is the pre-compression data rate being transferred over the network.
    - **Entries** — The current total number of entries in the BAB awaiting transmission.
    - **Percent BAB full** — The percentage of BAB in use by pending entries awaiting transfer.
    - **Percent done** — The percentage of data from the dtc devices transmitted over the network during a refresh or backfresh operation.
    - **Read KBps** — The local dtc device read I/O rate for applications accessing the dtc device.
    - **Write KBps** — The local dtc device write I/O rate for applications updating the dtc device.

    The measurements you can select for the secondary system are:
    - **Actual KBps** — The actual number of KBytes of data sent over the network.

- **Effective KBps** — If you have compression activated with the COMPRESSION tunable parameter, the effective KBps is the pre-compression data rate being transferred over the network.
- **Entry age recvd** — The age of the oldest entry received in seconds.

Click the **Modify Measurement** or **Remove From Chart** buttons to make changes.

Step III: Chart Display

When you have made all the necessary choices to define the chart you want, click the **Display Chart** button to display the chart on the screen.

NOTE Because of the fixed width of the chart label at the bottom of the display, if you size the chart and make it too narrow, the display becomes corrupted. Click the border of the chart window and drag the window larger to correct the problem.

Modifying a Chart

At this point, you can still change what's displayed in the chart. There are two ways to edit a chart:

- Left-click anywhere in the chart.

  The background color of the chart flashes briefly to yellow.

- Right-click anywhere in the chart.

  The **Chart Options Menu** is displayed. Select **Edit Chart**.

Modify the chart definition as desired and then select **Display Chart** to see the new version.

Other Chart Functions

You can display multiple charts at one time. Each chart is defined separately.

To delete a chart, print a chart, or save it in a file, use the **Delete Chart, Print Chart**, or **Print Chart to File** option from the **Chart Options Menu**. You can also delete a chart by selecting it to edit and then selecting **Delete Chart** in `dtcperftool`.

The Performance Tool displays error messages in red in the upper left area.

Performance Monitoring Files

`dtcperftool` obtains current performance information from a set of ASCII files created by **`throtd`** (RMDs). These files are called *performance monitoring files* or *performance tracking files*. These files are written in row/column format and are suitable for use by other applications such as spreadsheets or databases. They are located at:

- `/var/opt/SFTKdtc/p###.prf`
- `/var/opt/SFTKdtc/p###.prf.1` (previous generation file for this logical group)
- `/var/opt/SFTKdtc/p###.phd` (column descriptions for `.prf` files)

NOTE **For AIX only**
Replace `/var/opt/SFTKdtc` with `/var/dtc`.

Related Tunable Parameters

The following tunable parameters for `throtd` affect the performance monitoring files:

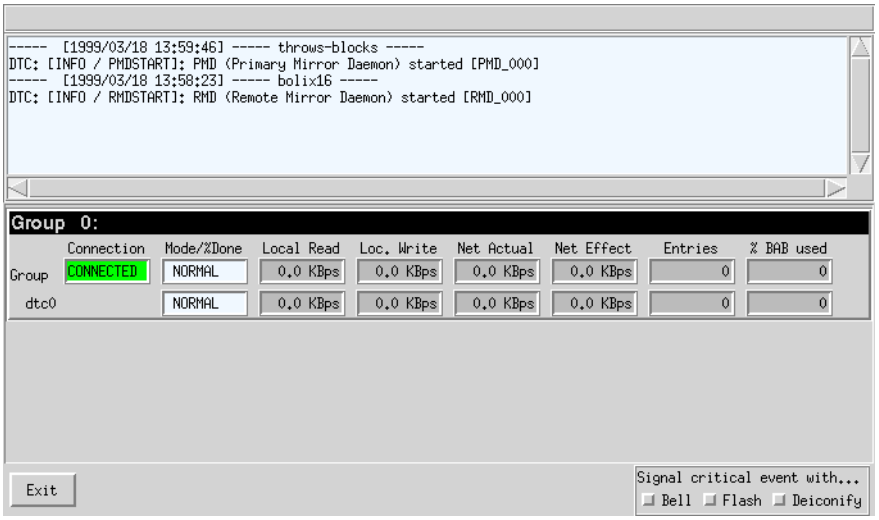
- STATINTERVAL
- MAXSTATFILESIZE

As `throtd` takes new performance snapshots, the latter are appended to the end of the appropriate performance monitoring file. Thus, a performance monitoring file records observations moving from the oldest at the top of the file to the most recent at the bottom of the file.

dtcmonitortool

`dtcmonitortool` provides a comprehensive picture of Softek Replicator activity and state information. It should be run only on the primary system.

dtcmonitortool Window



As the figure: *dtcmonitortool Window* shows, this utility display the following items:

- Status Message Area
- Error and Warning Messages
- Logical Group / dtc Device Status area
- Notification and Update Controls

Status Message Area

This area is at the very top of the window. If no Softek Replicator devices have been defined or initialized on the primary system, this status message area displays a message to that effect. The status message area displays `Updating...` while the status update information is being obtained.

Error and Warning Messages

Under the Status Message Area is a window that contains Softek Replicator error and warning messages listed from oldest at the top to the newest at the bottom. Each update cycle obtains new error or warning messages from the primary system and the secondary systems associated with it. Each message shows:

- The date and time that the message was generated
- Function name where the message originated
- Message level, message ID, message text

Messages are shown as follows:

- Fatal error messages — red
- Warning error messages — black

The size of this scrolling list is 200 messages by default; you can change it with the `dtcmonitortool` command line argument:

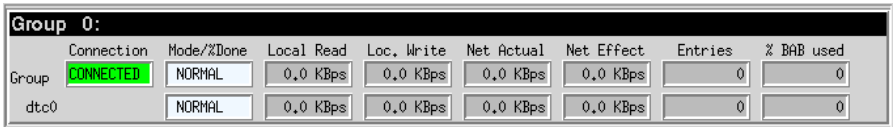
```
-messages count
```

Logical Group / dtc Device Status area

Below the Error and Warning Messages area is the Logical Group / dtc Device Status area. If this area of the window is not big enough to hold the status groups and status bars, then it automatically becomes a scrollable subwindow.

Each logical group defined on the primary system is shown in a bordered box as shown in the following figure. The top line indicates the group number (for example, **Group 0**) followed by the field descriptor.

Softek Replicator Group and Device Status Area of dtcmonitortool



The fields shown for the logical group display the following values:

- **Connection** — The state of the logical group's connection.
  - [CONNECTED] — (green) PMD and RMD daemons for this logical group are connected and active and will transfer any entries in the BAB.
  - [ACCUMULATE] — (red) Neither the PMD or RMD daemons for this logical group are present. Entries added to the local device are accumulating in the BAB.
  - [PMD ONLY] — (yellow) The PMD daemon alone is active and is attempting to create a connection with the RMD on the secondary system. Entries added to the local device are accumulating in the BAB.
  - [RMD ONLY] — (yellow) The RMD daemon for the logical group on the secondary system is active without a PMD on the primary system and should timeout and die within 30 seconds from the appearance of this indicator.

- **Mode % Done** — Softek Replicator's current operating state and percentage complete for refresh operations.
  - REFRESH — % Done
  - TRACKING
  - NORMAL
  - BACKFRESH — % Done
  - PASSTHRU
- **Local Read, Loc. Write** — Local reads and writes in KB per second.
- **Net Actual, Net Effect** — The actual rate of data flow (amount of data flow without compression) and the effective data rate (amount of data flow with compression and smart refresh taken into consideration) over the network.
- **Entries, % BAB used** — The number of entries in the BAB for each device and the percentage of BAB the device entries are consuming.
  - Grey — No entries in the BAB.
  - Green — Percentage in use is between 1 and 50.
  - Yellow — Percentage is between 51 and 80.
  - Red — Percentage is 80 or more.

Under the **Group** status is a status line for each dtc device defined for the logical group. On the figure on page 91, the shown status line is for **dtc0**. The items displayed are the same as those displayed for the logical group, but the values may vary.

Notification and Update Controls

At the bottom of `dtcmonitortool` are the **Notification** controls:

You can set these controls to occur if any indicators go into a "red" state, including receiving Fatal error messages.

dtcinfo

The `dtcinfo` command generates an ASCII report for one or more dtc devices indicating the Softek Replicator operating mode and performance metrics specific to the BAB

Run the `dtcinfo` command on the primary system only. The following figure shows a sample of the `dtcinfo` output.

dtcinfo Output Example

```
Requested BAB size ................ 134217728 (~
128 MB)
Actual BAB size ................... 102760448 (~
98 MB)
Free BAB size...................... 102760448
(~98 MB)

Logical Group 100 (throws-blocks -> plays-in-
traffic)

    Mode of operations.............. Passthru
    Entries in the BAB......... 0
    Sectors in the BAB......... 0
    Sync/Async mode................. Async
    I/O delay....................... 0

    Device /dev/dtc/lg100/rdsk/dtc0:

       Local disk device number........ 0x800179
        Local disk size (sectors)....... 263440
        Local disk name.................
        Remote mirror disk.........
        Read I/O count.................. 0
        Total # of sectors read......... 0
        Write I/O count................. 0
        Total # of sectors written...... 0
        Entries in the BAB......... 0
        Sectors in the BAB......... 0
```

6

Commands

dtcautomount 97

dtcbackfresh 98

dtccheckpoint 99

dtcconfigtool 100

dtcdebugcapture 101

dtchostinfo 101

dtcinfo 101

dtcinit 103

dtckillbackfresh 103

dtckillpmd 104

dtckillrefresh 104

dtckillrmd 105

dtclicinfo 105

dtcmonitortool 106

dtcoverride 106

dtcpmd 107

dtcperftool 107

dtcrefresh 108

dtcrmdreco 109

dtcset 110

dtcstart 111

dtcstop 112

dtcusersguide 112

killbackfresh 112

killdtcmaster 113

killpmds 113

killrefresh 114

killrmds 115

launchbackfresh 115

launchdtcmaster 116

launchpmds 116

launchrefresh 117

This chapter describes all commands of Softek Replicator.

NOTE You must be a super-user to execute commands.

dtcautomount

When the local data device is registered in FSTAB file as auto-mount, this command will release the mount for the local data device and register the dtc device mount information for the local data device to the FSTAB file.

NOTE The FSTAB file in Solaris is /etc/vfstab, and in HP-UX and Linux it is /etc/fstab, and in AIX it is /etc/filesystems.

Options:

-c

It checks the necessity of carrying out automatic mount registration of the DTC device, and result is outputted to standard output.

-g <group#>

Select one logical group.(essential)

-h

Displays help

Sample Input:

```
myserver# dtcautomount -g 101
```

Sample Output:

```
fstab was updated
```

NOTE **FSTAB is only updated if the following conditions are true:**

- AIX

When the data device of the specified logical group exists; the mount stanza of that record is defined to `[true] [automatic]`; and the dev stanza or the log stanza agrees with the device name.

- HP-UX

When the data device of the specified logical group exists, and the file system type of that record is other than `[swap] [swapfs] [dump] [ignore] [nfs]`.

- Linux

When the data device of the specified logical group exists, and the file system type of that record is other than `[swap] [proc] [tmpfs] [devpts] [dump] [ignore] [auto] [nfs]`.

- Solaris

When the data device of the specified logical group exists, and the automount is specified to `yes`.

During the update process, any data that is changed is first converted into a comment line and (Replicator-specific) character string, and the date of the update is added to the end of the record.

dtcbackfresh

This command initiates a backfresh operation as a means to synchronize data on the primary system with more up-to-date data on the secondary one.

NOTE BACKFRESH MODE is a maintenance mode only. Do not access or update local or mirror data devices until the backfresh is complete.

Options:

`-a`

Indicates all logical groups.

`-g <group#>`

Selects one or more logical groups to place into BACKFRESH MODE.

Sample Input:

```
myserver# dtcbackfresh -g 101
```

Output:

None.

dtccheckpoint

This command turns checkpointing of the mirror devices on or off. When checkpointing is on, Softek Replicator relinquishes its exclusive lock on the mirror device(s) and allows other applications to access the device(s) in read-only mode.

The `dtccheckpoint` command looks in the following directories for the optional, user-defined checkpoint scripts.

Primary scripts:

- `/etc/opt/SFTKdtc/cp_pre_on_p###.sh`
- `/etc/opt/SFTKdtc/cp_post_on_p###.sh`

Secondary scripts:

- `/etc/opt/SFTKdtc/cp_post_on_s###.sh`
- `/etc/opt/SFTKdtc/cp_pre_off_s###.sh`

NOTE `The SFTKdtc` directory does not exist on AIX systems; the `dtccheckpoint` command will look for optional, user-defined checkpoint scripts in the `dtc directory instead.`
For example: `/etc/opt/dtc/cp_pre_on_p###.sh`

If these scripts are implemented on the primary and secondary systems, they are executed when the command is entered. For more information on "checkpointing", refer to *Checkpointing Softek Replicator* on page 70.

Options:

`-a`

Indicates all logical groups.

`-g <group#>`

Selects one or more logical groups.

NOTE The `-p` and `-s` options for the `dtccheckpoint` command are only valid in a complex system configuration: when the system is set up to act as both primary and secondary systems.

`-p`

Causes the system on which the `dtccheckpoint` command is entered to act like a primary system and run the appropriate scripts.

`-s`

Causes the system on which the `dtccheckpoint` command is entered act like a secondary system and run the appropriate scripts.

`-on/off`

Turns "checkpointing" on or off.

Sample Input:

```
myserver# dtccheckpoint -g 101 -on
```

Output:

None.

dtcconfigtool

This command starts the utility for viewing, editing, or defining logical group configuration files, including primary and secondary systems, tunable PMD parameters, dtc devices, and throttles.

Options:

`-rmdtimeout t`

Sets the time (in seconds) during which the `dtcconfigtool` should wait for the RMD to return a list of disk partition devices and volumes defined on each secondary system. The default is 15.

`-noconfigchk`

Bypasses consistence and collision checking of component devices for dtc device definitions in each of the logical group configuration files when `dtcconfigtool` is brought up.

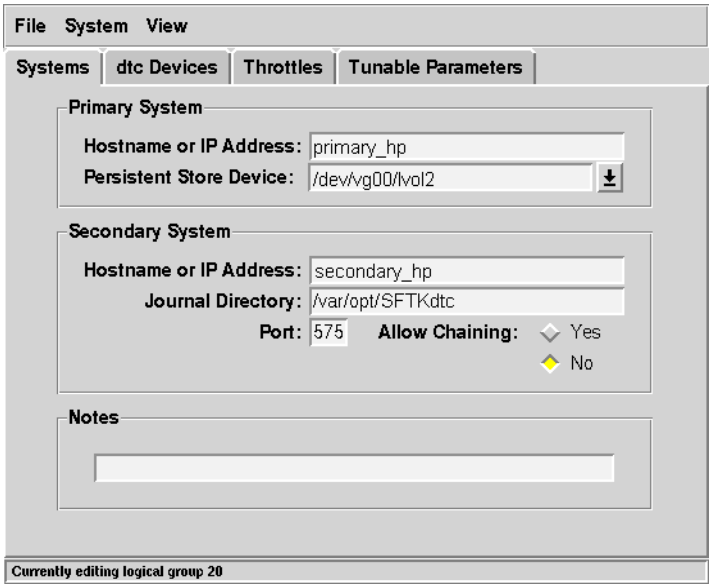
`-nohelp`

Suppresses the help messages that pop up after 2 seconds when the pointer is placed on a control or entry.

Sample Input:

```
myserver# dtcconfigtool
```

Sample Output:



dtcdebugcapture

This command collects system and software information that can be used for diagnosing problems with your Softek Replicator environment.

The information is saved in a file named `nodename1.tar.Z.uu`—for example, `onefish1.tar.Z.uu`—in directory `/tmp`. Save this file and send it to Softek Technical Support, if necessary.

Options:

None.

dtchostinfo

This command runs a utility that returns the networked hostnames and IP addresses for a system, and the host id of the system where the command is run.

You can run this command on either primary or secondary system to obtain local host information. From the primary system, you can also run the command with the secondary host name to obtain secondary system information.

Options:

`-hostname`

Specifies an optional hostname. Defaults to current hostname.

Sample Input:

```
myserver# dtchostinfo onefish
myserver# dtchostinfo <secondaryhostname>
```

(from the primary system only)

Sample Output:

```
193.91.182.100 onefish
hostid:0x82425e9b
```

dtcinfo

This command displays state information for one or more logical groups and their dtc devices. It must be run on the primary system.

The following state information can be displayed:

- BAB memory size requested and BAB memory actually allocated
- Logical Group operating mode (for example, NORMAL or PASSTHRU)
- dtc device information
- Version of the product installed on a system

Options:

`-g <group#>`

Displays information for all devices within the specified group. This option may be repeated to include more than one logical group.

`-a`

Displays information pertaining to all dtc devices for all started logical groups.

`-v`

Displays the version of the product installed on the system. This option can be used on the secondary system.

`-h`

Displays help.

Sample Input:

```
myserver# dtcinfo -g 0
```

Sample Output:

```
Requested BAB size ................ 67108864
(~ 64 MB)
Actual BAB size ......................
33554432 (~ 32 MB)
Free BAB size.........................
33554432 (~ 32 MB)

Logical Group 0 (pig -> pig)

   Mode of operations.............. Backfresh
    Shutdown state...................... 1
    Entries in the BAB................. 0
    Sectors in the BAB................ 0
    Sync/Async mode.................. Async
    I/O delay.............................. 0
    Persistent Store................... /dev/
dtcps

    Device /dev/dtc/lg0/rdsk/dtc0:

        Local disk device number........
0x4000000b
        Local disk size (sectors).........
131072
      Local disk name.....................
/dev/rdata1
        Remote mirror disk.................
pig:/dev/rdata2
       Read I/O count.....................
0
        Total # of sectors read............ 0
      Write I/O count.....................
0
        Total # of sectors written......... 0
       Entries in the BAB.................. 0
        Sectors in the BAB................. 0
```

NOTE For Logical groups in BACKFRESH mode, Shutdown state 0 means that the Logical Group is up and running. 1 means that the Logical Group is not up and running. If Shutdown state is 1 and the Logical Group needs to be operated, dtcstart should be executed.

dtcinit

This command initializes or resizes the BAB, or initializes the Pstore device. Using this command is the only way to specify a size other than the standard size available in the Configuration Tool dialog box.

Options:

`-b <bab_size_MB>`

Resizes the BAB and changes the `.cfg` file to reflect the designated size. `<bab_size>` should be specified in Megabytes as an integer between 1 and 1547.

Example:

```
dtcinit -b 512 MB
```

Sample Input:

```
myserver# dtcinit -b <bab_size_MB>
```

Output:

None.

`-p <pstore_device>`

Initializes the designated device to 0. If the specified device is used as a Pstore for an operating logical group, or the specified device is not defined in any `.cfg` files, the specified device is not initialized and the `dtcinit` command terminates.

Sample Input:

```
myserver# dtcinit -b 512
```

Output:

None

dtckillbackfresh

This command terminates the BACKFRESH MODE for one or more logical groups.

`dtckillbackfresh` moves the target logical groups out of BACKFRESH MODE and into NORMAL MODE. This action takes place whether the PMD is running or not.

If the target PMD is not running but is in BACKFRESH MODE (that is, if the PMD was in BACKFRESH MODE and the `killpmds` command was entered for it), then when it is restarted with the `launchpmds` command, it is restarted in NORMAL MODE.

Options:

`-g <group#>`

Terminates backfresh daemon for logical group <group#>. Can be repeated to include multiple logical groups.

-a

Terminates all backfresh daemons.

-h

Displays help.

Sample Input:

```
myserver# dtckillbackfresh -a
```

Output:

None.

dtckillpmd

This command terminates one or more active PMD daemons.

NOTE If any target PMD is in BACKFRESH MODE when dtckillpmd is entered for it, then Softek Replicator will remember that; when the PMD is restarted, Softek Replicator restarts it in BACKFRESH MODE where it left off. The only way to take the PMD out of BACKFRESH MODE is with the killbackfresh command.

Options:

-g <group#>

Terminates PMD daemon for logical group <group#>. This option can be repeated to affect multiple groups.

-a

Terminates all PMD daemons.

-h

Displays help.

Sample Input:

```
myserver# dtckillpmd -a
```

Output:

None.

dtckillrefresh

This command terminates REFRESH MODE in one or more logical groups.

Options:

-g <group#>

Terminates REFRESH MODE for logical group <group#>. Can be repeated to include multiple logical groups.

-a

Terminates REFRESH MODE for all logical groups.

`-h`

Displays help.

Sample Input:

```
myserver# dtckillrefresh -a
```

Output:

None.

dtckillrmd

This command terminates one or more active RMD daemons.

Options:

`-g <group#>`

Terminates RMD daemon for logical group `<group#>`. This option can be repeated for multiple groups.

`-a`

Terminates all RMD daemons.

`-h`

Displays help.

Sample Input:

```
myserver#  dtckillrmd -a
```

Output:

None.

dtclicinfo

This command reports the state of Softek Replicator licenses on this system. It also alerts you if the license is missing or expired.

Options:

None.

Sample Input:

```
myserver# dtclicinfo
```

Sample Output:

```
Valid DTC Demo License -- expires: Mon May 31 15:30:40 1999
```

dtcmonitortool

This command starts a tool for displaying Softek Replicator performance statistics in real time. `dtcmonitortool` must be run on the primary, and only monitors logical groups that have been started. It displays current values for a variety of parameters, error messages from both primary and secondary systems, and alerts the operator of potentially fatal errors.

Options:

`-timeout seconds`

Sets the number of seconds during which `dtcmonitortool` waits for a connection to be established to a primary or secondary system to get the latest error messages and daemon status. Default: 30.

`-messages count`

Sets the maximum number of retained error and warning messages. Default: 200.

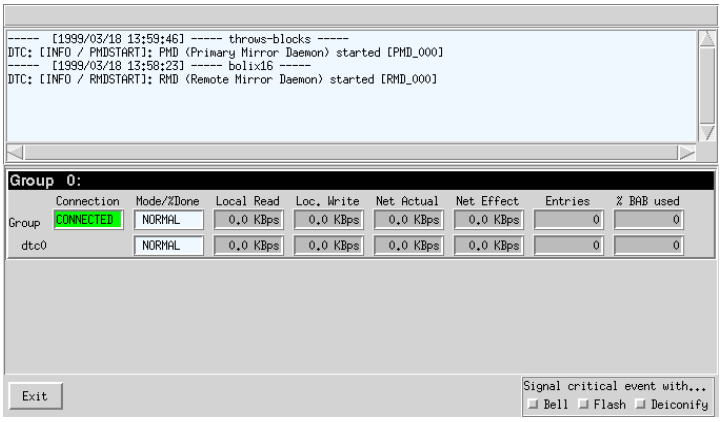
`-help`

Displays the command line argument help message and exits.

Sample Input:

```
myserver# dtcmonitortool
```

Sample Output:



dtcoverride

The `dtcoverride` command clears the BAB or pstore or forces a transition between Softek Replicator operating modes.

NOTE Use caution when issuing this command, since it may cause a loss of synchronization between the systems in Softek Replicator.

Options:

`-a`

Validates all logical groups to be affected by the `state` forced change of state.

`-g <group#>`

Selects one or more logical groups to be affected by the `state` forced change of state.

`clear [BAB|LRT|HRT]`

The `BAB` option clears the BAB. The `LRT` and `HRT` options clear the tracking bitmaps.

`state [state]`

Forces a change of state for the designated logical groups. The states you can specify are: `passthru`, `tracking`, and `normal`.

Note that if you specify a state of `tracking` or `passthru` on the `dtcoverride` command, the PMD daemon associated with the logical group is killed (if it is running) before the change of state takes effect.

Sample Input:

```
myserver# dtcoverride -g 101 state normal
```

Output:

None.

dtcpmd

This command starts PMD daemons for one or more logical groups.

Options:

`-g <group#>`

Starts PMD daemon for logical group `<group#>`. This option can be repeated for multiple groups.

`-a`

Starts all PMD daemons for all logical groups on this system.

`-h`

Displays help.

Sample Input:

```
myserver#  dtcpmd -a
```

Output:

None.

dtcperftool

Thiscommand starts the graphical charting tool for displaying Softek Replicator performance data

Options:

If primary and secondary configuration files are present on the system when `dtcperftool` is run, use one of the following arguments to determine which files are queried:

`-p`

Specifies that `dtcperftool` should look for primary configuration files only.

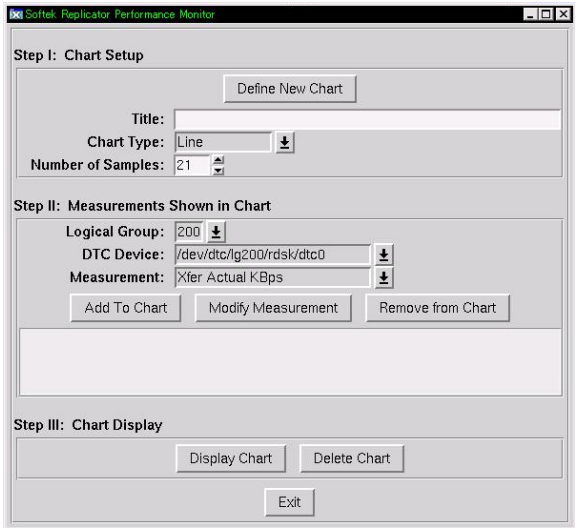
`-s`

Specifies that `dtcperftool` should look for secondary configuration files only.

Sample Input:

```
myserver# dtcperftool
```

Sample Output:



dtcrefresh

This command places one or more logical groups in REFRESH MODE to synchronize the secondary system's mirror devices with the contents of the primary system's local data devices. Use `dtcrefresh` to:

- Establish an initial remote mirror during a new Softek Replicator installation.
- Reestablish a remote mirror if the BAB fills and automatically moves the logical groups into TRACKING MODE on the primary system by transferring only the changed data (smart refresh).
- Reestablish the mirror after a remote mirror device is replaced due to hardware failure.

NOTE Use the `-f` option to perform a full, sector by sector synchronization of the systems.

Options:

`-g <group#>`

Places all dtc devices in logical group `<group#>` in REFRESH MODE. This option can be repeated for multiple groups.

`-a`

Places all dtc devices for all groups in REFRESH MODE.

`-c`

Initiates a checksum refresh in which all blocks on the local data device and mirror device are compared using a checksum method to identify deltas. Only blocks that have been modified (that is, those for which the checksum varies) are sent to the secondary system. A checksum refresh writes mirror data to the journal file system, not directly to the mirror device.

`-f`

Forces a full refresh of all data blocks from the local data devices to the mirror devices on the secondary system.

Sample Input:

```
myserver# dtcrefresh -a
```

Output:

None.

dtcrmdreco

This command should be used only when you are switching data ownership over to the secondary system. It is a recovery utility that ensures the mirror devices are in a coherent and recoverable state by flushing data from the journal file(s) to the corresponding mirror device(s).

The `dtcrmdreco` command creates a `/etc/opt/SFTKdtc/s###.off` file for each logical group. When the primary system comes back online or when a new system is added to the configuration, the logical group's RMD daemon detects the `s###.off` file and does not start mirroring operations. This keeps the mirror devices from being corrupted before you perform a backfresh/refresh.

NOTE On AIX, the `dtcrmdreco` command will create a `s###.off` file in the `/etc/dtc/lib/` directory.

See *Recovering Data* on page 79 for detailed information on using this command for data recovery.

Options:

`-g <group#>`

Recovers data for logical group `<group#>`.

`-a`

Recovers data from all logical groups.

NOTE One of the above-mentioned options MUST be used in the `dtcrmdreco` command.

`-d`

Deactivates the recovery mode.

NOTE The `-d` option must only be used AFTER the recovery has been performed for the group. Using the `-d` option before will generate the following error:
`Couldn't unlink file /etc/opt/SFTKdtc/s###.off: The system cannot find the file specified.`

Sample Input:

On the secondary:

```
dtcrmdreco -g0
```

Activates the recovery mode and creates the `s###.off` file.

```
dtcrmdreco -d -g0
```

Deactivates the recovery mode and removes the `s###.off` file.

Then on the primary:

```
launchbackfresh -g0
```

Output:

None.

dtcset

This command sets tunable parameters for each designated logical group. You can also use it to view the current setting of a specific tunable parameter, or all tunable parameters for a logical group. Run it on the primary system only.

Note that you must run `killpmds` before and `launchpmds` after changing the following tunable parameters:

- CHUNKSIZE
- COMPRESSION
- NETMAXKBPS

You must also run `dtcstop` before and `dtcstart` after changing the following tunable parameters:

- SYNCMODE
- SYNCMODEDEPTH
- SYNCMODETIMEOUT

Options:

`-g <group#> <keyword>=<value>`

Sets value of designated tunable parameter for each logical group. If you do not specify a keyword or value, Softek Replicator shows you all tunable parameter values for the logical group.

Sample Input:

```
myserver# dtcset -g 6 netmaxkbps=500
```

Sample Output:

```
myserver# dtcset -g 6
netmaxkbps=500
myserver# dtcset -g 6
CHUNKSIZE: 2048
CHUNKDELAY: 0
SYNCMODE: off
SYNCMODEDEPTH: 1
SYNCMODETIMEOUT: 30
NETMAXKBPS: 500
STATINTERVAL: 10
MAXSTATFILESIZE: 64
TRACETHROTTLE: off
COMPRESSION: off
myserver#
```

dtcstart

The `dtcstart` command processes the configuration file (`.cfg`) for the logical group(s) specified and activates the dtc devices defined within the group. As the command processes each file, it creates a copy that it renames with a `.cur` extension. The `.cur` file is an exact copy of the group's configuration file when it was started and is referenced by all Softek Replicator commands during operations

Configuration(`.cfg`) files are only processed by `dtcstart`, so when a group is stopped and restarted a new `.cur` file is created. This allows you to make modifications to the `.cfg` files (for example, edit throttle definitions) and have precise control over those changes when they are implemented. Also, the `.cur` file ensures that all components of Softek Replicator have a consistent view of the configuration.

Starting a logical group make its dtc devices become available in the dropdown menus in the Configuration Tool.

NOTE Running `dtcstart` on a logical group with a large number of dtc devices can be slow.

Options:

`-g <group#>`

Starts a specified logical group and its dtc devices.

`-a`

Starts all logical groups and their dtc devices.

`-b`

(For boot scripts only) restarts previously started logical groups and their dtc devices that were active prior to a system crash or that were stopped with the `dtcstop -s` option.

Sample Input:

```
myserver# dtcstart -a
```

Sample Output:

None.

dtcstop

This command stops one or all logical groups and corresponding dtc device definitions. The tracking bitmaps are written to the pstore and the dtc device definitions associated with the logical group(s) are removed from the `/dev/dtc` device tree, making them no longer available for active use.

Options:

`-g <group#>`

Stops a specified logical group and removes its dtc devices.

`-a`

Stops all logical groups and removes their dtc devices.

`-s`

(For boot scripts only) Stops the previously started logical groups but marks them enabled for restart when the system is next booted.

Sample Input:

```
myserver# dtcstop -a
```

Sample Output:

None.

dtcusersguide

This shell script opens the product documentation in Adobe Acrobat.

Options:

None.

killbackfresh

Shell script that runs `dtckillbackfresh`; terminates any backfresh operation currently running on this system.

`killbackfresh` provides the following additional functionality beyond that provided by `dtckillbackfresh`.

- The script checks to see if any logical group PMDs are running.
- If a PMD daemon is running, then `dtckillbackfresh` is started.
  - If command line arguments are specified, they are passed to `dtckillbackfresh`.
  - If no command line arguments are specified, then `dtckillbackfresh` is run with the `-a` argument, which terminates all logical group PMDs that are currently in BACKFRESH MODE. A message is written to the console stating that the backfresh operations have been terminated.
- If there is no PMD daemon running, then a message is written to the console saying that there is no PMD daemon running, and the script exits.

Options:

`-g <group#>`

Terminates backfresh daemons for the logical group `<group#>`. This option can be repeated for multiple logical groups.

```
killbackfresh -g 3 -g 12
```

`-a`

Terminates all backfresh daemons.

Sample Input:

```
myserver# killbackfresh -a
```

Output:

If the script completes and the command is executed, there is no output. If there is no PMD daemon running, a message indicating this is written to the console and the script exits.

killdtcmaster

Shell script that terminates any PMD daemon, RMD daemon, `in.dtc`, or `throtd` processes currently running on the system. Use this script for maintenance activities such as stopping any logical groups with the `dtcstop` command, or removing Softek Replicator from a system prior to a software upgrade.

Options:

None.

Sample Input:

```
myserver# killdtcmaster
```

Sample Output:

```
in.dtc master Softek Replicator daemon has been shutdown
throtd Softek Replicator throttle daemon has been shutdown
```

killpmds

Shell script that runs `dtckillpmd`, terminating PMD daemons currently running on this system. When PMD daemons are terminated, their corresponding RMD daemons on the secondary systems are terminated as well.

`killpmds` provides the following additional functionality beyond that provided by `dtckillpmd`:

- The script checks to see if any logical group PMD daemons are running.
- If PMD daemons are running, then `dtckillpmd` is started.
  - If command line arguments are specified, they are passed to `dtckillpmd`.
  - If no command line arguments are specified, then `dtckillpmd` is started with the `-a` argument, which terminates all PMD daemons.
- If there is no PMD daemon running, then a message is written to the console saying that there is no PMD daemon running, and the script exits.

Options:

`-g <group#>`

Terminates PMD daemons for logical group `<group#>`. This option can be repeated for multiple groups.

```
killpmds -g 12 -g 0
```

`-a`

Terminates all PMD daemons.

Sample Input:

```
myserver# killpmds
```

Output:

If the command is executed, there is no output. If there are no PMD daemons running, a message indicating this is written to the console and the script exits.

killrefresh

Shell script that runs `dtckillrefresh`; terminates any refresh operations currently running on this system.

`killrefresh` provides the following additional functionality beyond that provided by `dtckillrefresh`:

- The script determines whether any logical group refresh operations are running.
- If refresh operations are running, then `dtckillrefresh` is started.
  - If command line arguments are specified, they are passed to `dtckillrefresh`.
  - If no command line arguments are specified, then `dtckillrefresh` is run with the `-a` argument, which terminates all logical group refresh operations.
- If there are no refresh operations running, then a message is displayed that there are no refresh operations running, and the script exits.

Options:

`-g <group#>`

Terminates REFRESH MODE for logical group `<group#>`. This option can be repeated for multiple logical groups.

```
killrefresh -g 2 -g 10
```

`-a`

Terminates REFRESH MODE for all logical groups.

Sample Input:

```
myserver# killrefresh -a
```

Output:

If the script completes and the command is executed, there is no output. If no refresh operations are running, a message indicating this is written to the console and the script exits.

killrmds

Shell script that terminates all RMD daemon currently running on this system. Enter the command on the secondary system.

Once the RMD daemon has been killed, you can restart it by either killing (`killpmds`) and relaunching (`launchpmds`) the associated PMD daemon on the primary system.

`killrmds` provides the following additional functionality beyond that provided by `dtckillrmd`:

- The script checks to see if any logical group RMD daemons are running.
- If RMD daemons are running, then `dtckillrmd` is started.
  - If command line arguments are specified, they are passed to `dtckillrmd`.
  - If no command line arguments are specified, then `dtckillrmd` is started with the `-a` argument, which terminates all RMD daemons.
- If there are no RMD daemons running, then a message is written to the console saying that there are no RMD daemons running, and the script exits.

Options:

`-g <group#>`

Terminates the RMD daemon for logical group `<group#>`. This option can be repeated for multiple logical groups.

```
killrmds -g 2 -g 10
```

`-a`

Terminates the RMD daemon for all logical groups on this secondary system.

Sample Input:

```
myserver# killrmds
```

Output:

If there is no RMD daemon running, then a message indicating this is written to the console, and the script exits.

launchbackfresh

Shell script that runs `dtcbackfresh` to start backfresh operations. Recall that backfresh synchronizes the primary system with the secondary system using the mirrored data.

NOTE BACKFRESH MODE is a maintenance only mode. Do not access or update local or mirror data devices until the backfresh is complete.

`launchbackfresh` provides the following functionality beyond that provided by the `dtcbackfresh` command:

- The script launches the PMD daemons if they are not already running. It validates system license information and then starts `dtcbackfresh`.
- If no command line arguments are specified, then `dtcbackfresh` is run with the `-a` argument, which puts all logical groups and all devices into BACKFRESH MODE.

See *Recovering Data* on page 79 for more detailed information on using this command for data recovery.

Options:

`-a`

Indicates all logical groups.

`-g <group#>`

Selects one or more logical groups designated by the `<group#>`.

Sample Input:

```
myserver# launchbackfresh -a
```

Output:

None.

launchdtcmaster

Shell script that starts the master Softek Replicator daemon (*in.dtc*) and Softek Replicator throttle daemon (`throtd`). This is a convenience utility to reestablish the Softek Replicator daemon environment after system maintenance when you entered the `killdtcmaster` command. The Softek Replicator master daemon and throttle daemon are launched automatically during installation and after reboot. You do not normally use this command except for system maintenance activities.

Sample Input:

```
myserver# launchdtcmaster
```

Output:

None.

launchpmds

Shell script that starts PMD daemons for the designated groups. This is the preferred method of starting the daemons.

`launchpmds` provides the following additional functionality beyond that provided by `dtcpmd`:

- The script checks that a valid license file (`DTC.lic`) is present on the system.
- The script checks to see if the PMD daemons are already running.
- If the PMD daemon is not running, then `dtcpmd` is started.
  - If command line arguments are specified, they are passed to `dtcpmd`.
  - If no command line arguments are specified, then `dtcpmd` is started with the `-a` argument, which starts all PMD daemons.
- If there are PMD daemons running, then a message is written to the console saying that there is a PMD daemon running, and the script exits.

Options:

`-g <group#>`

Starts the PMD daemon for logical group <group#>. This option can be repeated for multiple groups.

-a

Starts PMD daemons for all logical groups on this system.

Sample Input:

```
myserver#  launchpmds -g 1 -g 2
```

Output:

None.

launchrefresh

Shell script that starts dtcrefresh, which causes the product to transition to REFRESH MODE and synchronize the local data device contents with the mirror devices. This is the preferred method for starting dtcrefresh.

launchrefresh provides the following additional functionality beyond that provided by dtcrefresh:

- If no command line arguments are specified, then dtcrefresh is run with the -a argument, which starts a smart refresh operation on all logical groups.
- If a full refresh is required, use the -f option.

Options:

-g <group#>

Puts all dtc devices in logical group <group#> into REFRESH MODE. This option can be repeated for multiple groups.

```
launchrefresh -g 1 -g 21
```

-a

Puts all dtc devices for all groups into REFRESH MODE.

-c

Initiates a checksum refresh in which all blocks on the local data device and mirror device are compared using a checksum method to identify deltas. Only blocks that have been modified (that is, those for which the checksum varies) are sent to the secondary system. A checksum refresh writes mirror data to the journal file system, not directly to the mirror device.

-f

Forces a full refresh of all data blocks from the data devices to the mirror devices on the secondary system.

Sample Input:

```
myserver# launchrefresh -a
```

Output:

None.

7

Complex Configurations

Symmetric Configuration 121

One-to-many 125

Chaining 127

Loopback Configuration 129

This chapter explains the processes involved to configure Softek Replicator in environments that are slightly more complex than a single primary to a secondary configuration.

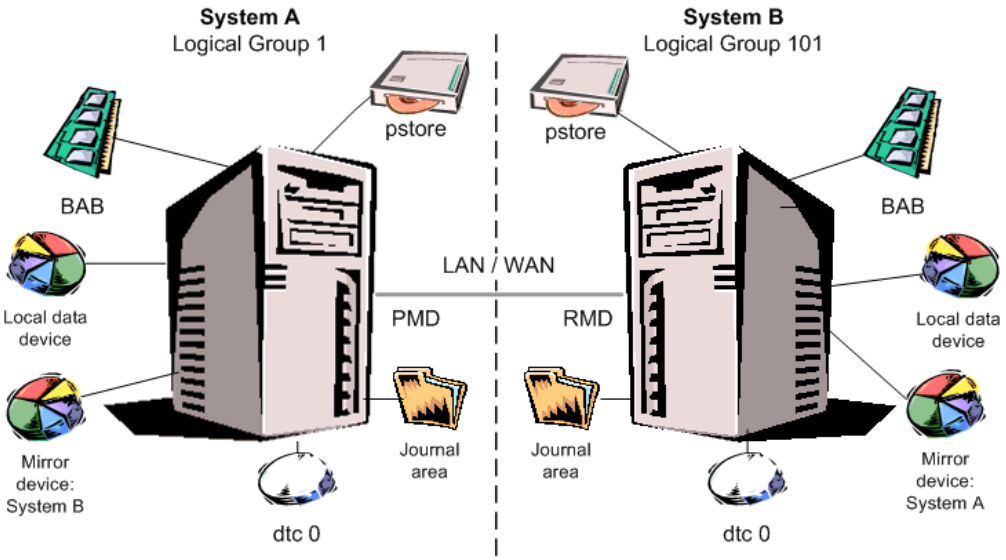
Symmetric Configuration

A symmetric configuration is appropriate for the enterprise in which an automated switch-over is desired for load-balancing or other planned events. Two-way mirroring is not achieved in this manner. Rather, data is being mirrored from the primary system (A) to the secondary system (B). The difference being that the secondary is symmetrically configured and ready to assume control of operations if application focus is deliberately moved to it. After this type of switch-over, the secondary system assumes the role of primary, and the previous primary system becomes the new secondary one.

When creating this type of configuration, it is best to plan out the logical groups and dtc devices for both systems prior to initiating the install. The key to this configuration is having the mirror device on the secondary be a dtc device rather than a raw disk or volume. There is, of course, a mirror device for each local data device on the primary, and each of these mirror devices should be a dtc device on the system B. Basically that means establishing a BAB and pstore on both systems in a symmetric configuration and using dtc device names for the mirror devices rather than `dsk` or `rdsk` conventions.

You should also establish B's mirror devices so they are the same as A's local data device. Thus, when a switch-over occurs, B is mirroring to the appropriate device and the data on both systems remains coherent and up to date. The following figure describes the relationship between devices in a symmetric configuration.

Sample Symmetric Configuration



As you can see, in a symmetric configuration, System A mirrors to System B and System B mirrors to System A.

NOTE The partitions used for the dtc devices must be identical in size.

Defining a Symmetric Configuration

On System B

1. Determine the systems configuration.
2. Establish the logical group(s) and dtc device numbers for each system. An effective methodology for defining this kind of symmetric configuration is to bias the logical group numbering by 100. The following table gives a sample configuration.

Sample Symmetric Definitions

Primary System (A)	Secondary System (B)
Logical Group 1	Logical Group 101
dtc device 0 (dtc0)	dtc device 0 (dtc0)
local data device: `/dev/rdsk/c1t2d0s1`	local data device: `/dev/rdsk/c1t3d0s1`
mirror data device (same as the dtc device path for logical group 101 on System B): `/dev/dtc/lg101/rdsk/dtc0`	mirror data device (same as local device on System A): `/dev/rdsk/c1t2d0s1`

3. Run `dtcconfigtool` on System B to configure logical group 101 as described in the *Sample Symmetric Definitions* on page 122 table.

   Allocate the BAB and pstore, and set up the logical group(s) and dtc devices. Keep in mind that devices defined as mirror devices on System B need to be the same device name as the local data device on System A. For clarification, refer to *Sample Symmetric Configuration* on page 121.

4. **Commit** devices, save the logical group, then exit `dtcconfigtool`.
5. Copy configuration files over to System A and rename.
6. Verify that dtc devices are not mounted automatically when the system is booted.
7. Type `dtcstart -g 101` to start the group.
8. Run `dtcinfo -g 101` to verify that the group is in PASSTHRU MODE.

NOTE The logical group(s) on System B MUST be running in PASSTHRU MODE to allow A to B mirroring.

On System A

9. Run `dtcconfigtool` to configure logical group 1 as described in *Sample Symmetric Definitions* on page 122.
10. Allocate the BAB and pstore.

11. Define logical groups and dtc devices for the primary on A as in *Sample Symmetric Definitions* on page 122.

NOTE Remember that mirror devices for logical group 1 should be dtc device names for logical group 101. All started dtc devices are displayed in the drop-down menu in `dtcconfigtool`, and can be selected appropriately as mirror devices.

12. **Commit** Devices and exit.
13. Copy configuration files over to System B and rename.
14. Verify that dtc devices are not mounted automatically when the system is booted.
15. Type `dtcstart -g 1` on the primary system.
16. Softek Replicator is operating in PASSTHRU MODE at this point. A refresh operation or a manual operation is required to transition into NORMAL MODE. This initializes the mirror for logical group 1.

Running B as Stand-Alone

1. On System B type `dtcrmdreco -g 1`.

   To verify that logical group 1 is in recovery mode, check that the s001.off file is in the `/etc/opt/SFTKdtc` on B.

NOTE **For AIX only**
Replace `/etc/opt/SFTKdtc` with `/etc/dtc/lib`.

NOTE This step is necessary to prepare the mirror devices to be accessed as data devices. The `dtcrmdreco` command flushes all outstanding entries to disk and prohibits further mirroring to these devices.

2. On System A, type `dtcstop -a`.
3. On System B, place logical group 101 into TRACKING MODE by entering:
   `dtcoverride -g 101 state tracking`
4. `fsck` the logical group 101 dtc devices that had file systems mounted.
5. `mount` your file systems and start applications on System B's dtc devices.

NOTE Placing B into TRACKING MODE reduces the length of time required for a refresh operation when A returns to service.

Inverting the A-B Topology to B-A

On System A:

1. When System A boots, logical group 1 (and any other logical group defined on this system) is started in PASSTHRU MODE. Type a `dtcstop -a` to stop all logical groups on System A. The `-a` option inhibits these logical groups from being restarted on System A after each reboot.

On System B:

2. On System B `launchrefresh -g 101`. If System B was put into TRACKING MODE as illustrated above, this refresh should be very efficient since it requires only changed data be transferred to System A.

NOTE Applications may continue to update dtc devices in logical group 101 during the refresh.

3. Continue running the applications on System B until it is desirable to make System A the primary once again.

Switching Back from A to B

1. Halt the applications on System B and force all pending updates to disk (for example, with the **sync** command). Unmount any file systems on the dtc devices in logical group 101.
2. Verify that all updates have been mirrored to System A.
3. On System A, type `dtcrmdreco -g 101`. Then, verify that the s101.off file is in the `/etc/opt/SFTKdtc` directory.

NOTE **For AIX only**
Replace `/etc/opt/SFTKdtc` with `/etc/dtc/lib`.

4. On System B, type `dtcoverride -g 101 state passthru`.
5. Type the `dtcrmdreco -g 1 -d` command on B to enable mirroring to the dtc devices in logical group 101 on System B in an A-B topology.
6. Verify that the s001.off file is NOT in the `/etc/opt/SFTKdtc` directory on System B.

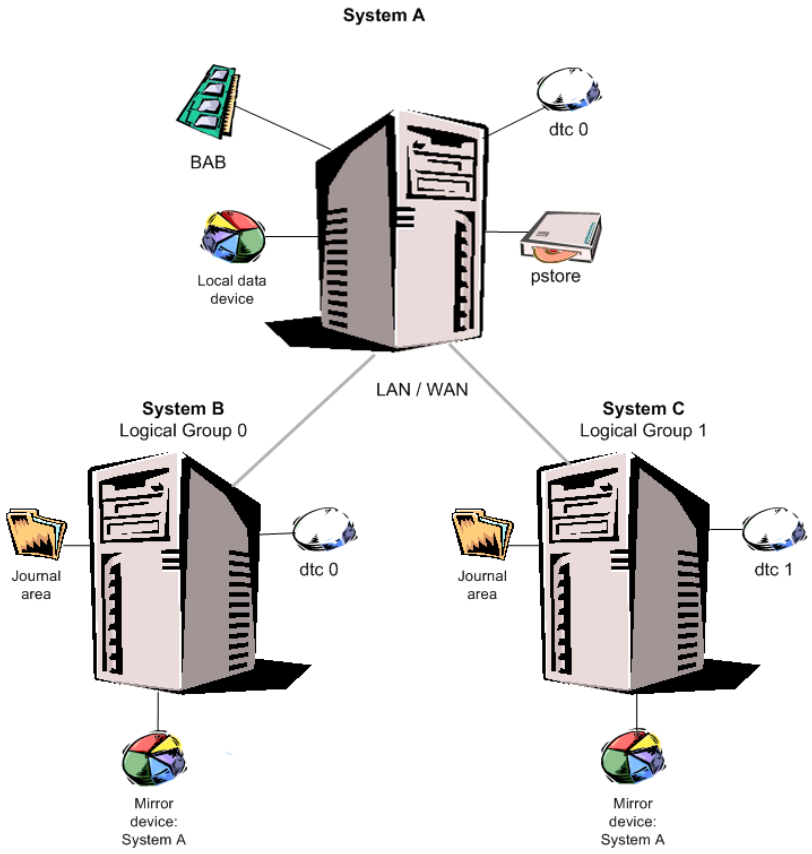
On System A:

7. Start logical group 1 with `dtcstart -g 1`.
8. Type `dtcoverride -g 1 state normal`.
9. Mount file systems and start applications.

One-to-many

This configuration allows the mirroring of the same data set to two or more secondary systems. The following figure shows the process for establishing a configuration where System A mirrors to System B while simultaneously mirroring the same data to System C.

Sample One-To-Many Configuration



Sample One-To-Many Definitions

Primary System (A)

Logical Group 0	**Logical Group 1**
dtc device 0 (dtc0)	dtc device 1 (dtc1) This dtc device maps to dtc0 as the local data device and to the raw device on System C.
local data device:	local data device:
mirror data device (on System B):	mirror data device (on System C):

The figure: *Sample One-To-Many Configuration* shows that System A is simultaneously mirroring the same data set to System B and System C.

▶ **To set up this type of configuration:**

1. Plan the devices to be used on all systems.
2. Run `dtcconfigtool` to allocate BAB on the primary (System A). Create logical group 0 which specifies System A as the primary and System B as the secondary.
3. Define dtc 0.
4. **Commit** devices, and Exit from the **File** menu.
5. Copy the .cfg file to B and rename to s000.cfg.
6. Start logical group 0 on System A with the `dtcstart -g 0` command.
7. Use `dtcconfigtool` to create a new logical group 1, with System A as the primary and System C as the secondary).
8. From the drop down list, select logical group 0's dtc 0 as the local data device.

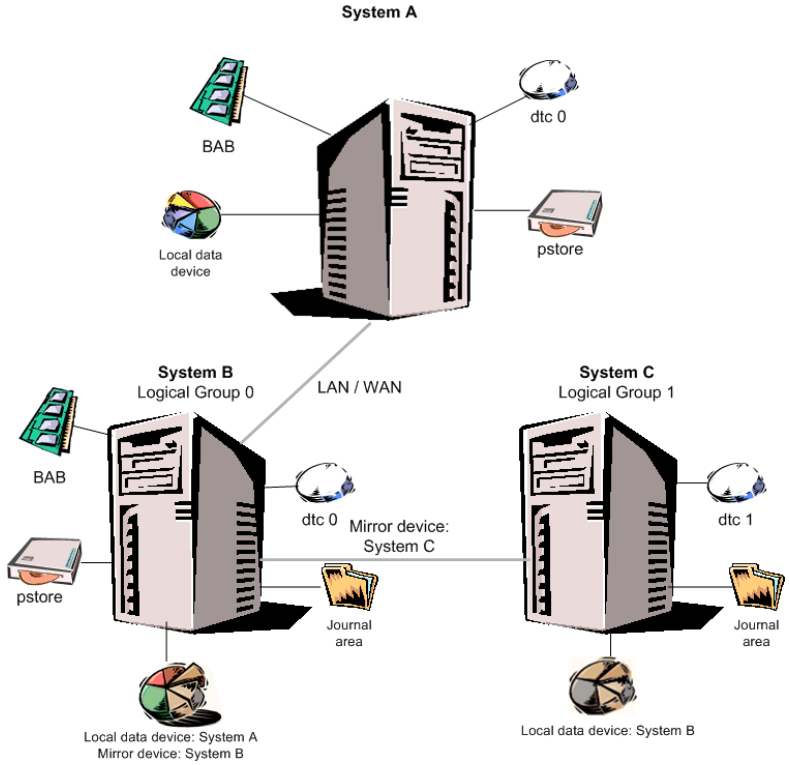
NOTE All dtc devices that have been started are visible and can be selected from the drop down menu in `dtcconfigtool`.

9. Commit the device. From the File menu, Exit. Save changes when prompted.
10. Copy the .cfg file to C and rename to s001.cfg.
11. Type `dtcstart -g 1`.
12. Launch a refresh on A using the `launchrefresh -af` command.
13. Mount file systems and have applications only interact with dtc devices defined in logical group 1. This provides mirroring to both B and C.

Chaining

Chaining refers to a configuration where systems appear to be connected in a sequential manner, such as A, B, and C where Systems A and C have no connection or awareness of the other. The following figure shows the process for setting up a configuration in which System A mirrors to System B, which, in turn, mirrors to System C.

Sample Chaining Configuration



The figure: *Sample Chaining Configuration* shows a configuration where System A is in the role of primary, mirroring to System B as a secondary, and System B is in the role of primary, mirroring to System C as a secondary.

The B to C component of this configuration is similar to a simple A to B setup with two minor variations. You should do the B to C part of the setup first.

NOTE When you define Sync mode in chain structure, please set the tunable parameter, SYNCMODETIMEOUT, bigger for A to B than for B to C.

Sample Chaining Definitions

Primary System (A)	Secondary System (B)
Logical Group 0	**Logical Group 100**

Sample Chaining Definitions

dtc device 0 (dtc0, dtc1...)	dtc device 0 (dtc0)
local data device:	local data device:
mirror data device (on System B):	mirror data device (on System C):
The mirror data device maps to the dtc device (s) defined for Logical Group 100.	

B to C Setup

1. Run `dtcconfigtool` and define Logical Group 100 (instead of defaulting to logical group 0) with System B as primary and System C as secondary.
2. Define dtc devices (dtc0, dtc1, etc.) as normal.
3. Copy the configuration file `/etc/opt/SFTKdtc/p100.cfg` to `/etc/opt/SFTKdtc/s100.cfg` on System C.

NOTE **For AIX only**
Replace `/etc/opt/SFTKdtc` with `/etc/dtc/lib`.

4. Use the `dtcstart -g 100` command to start the logical group.
5. Use the `dtcoverride -g 100 state normal` command to bypass the refresh operation and place the logical group into NORMAL MODE.
6. Execute `launchpmds` to establish a network connection between B and C.

NOTE Do not mount any file systems or have applications interact with the dtc devices defined in logical group 100 on System B!

A to B Setup

1. Use `dtcconfigtool` to create a configuration for logical group 0 that has System A as primary and System B as secondary.
2. Note that in the **Secondary** area, there is a box for turning chaining on or off. Select on (the default is off).
3. Define the dtc devices (dtc0, dtc1) but in the drop-down menu for the mirror devices, select the dtc device as defined for logical group 100 on System B. For example: `/dev/dtc/lg100/rdsk/dtc0`

   Repeat steps 1 to 3 for each dtc device in the logical group.
4. **Commit** device.
5. From the **File** menu, **Save** changes and **Exit** the tool.

6. Copy the configuration file `/etc/opt/SFTKdtc/p000.cfg` to `/etc/opt/SFTKdtc/s000.cfg` on System B.

NOTE **For AIX only**
Replace `/etc/opt/SFTKdtc` with `/etc/dtc/lib`.

7. Run `dtcstart -g 0` on System A to start the group.

8. Type a `launchrefresh -g 0 -f` on System A.

At this point the chaining topology is active and in effect. To verify this, during the refresh operation, run `dtcmonitortool` on System B to watch the refresh updates from System A arrive as input to the dtc devices defined in logical group 100.

Loopback Configuration

Softek Replicator supports a loopback configuration, that is, a configuration in which both primary and secondary systems are on the same system. This can be useful for a variety of needs, such as testing or employing "checkpointing" to create point-in-time snapshots of data sets. To define a loopback environment:

1. Launch `dtcconfigtool`.

2. Under the **Systems** tab, specify the same host name or IP address for both primary and secondary systems.

3. Define dtc devices as usual.

A

Configuration Files

Configuration File Sample of a Primary Logical Group 133

Configuration File Sample of a Primary Logical Group

Softek Replicator Configuration File: /etc/opt/SFTKdtc/p000.cfg

```
#=============================================
==========
# Softek Replicator Configuration File:   /etc/
opt/SFTKdtc/p000.cfg
#
#  Last Updated:  Mon Mar 08 13:02:09 MST 2000
#  Softek Replicator Version 2.1
#=============================================
==========
#
NOTES:  Softek Replicator www pages
#
#
# Primary System Definition:
#
SYSTEM-TAG: SYSTEM-A PRIMARY
  HOST: red
  PSTORE:
#
# Secondary System Definition:
#
SYSTEM-TAG: SYSTEM-B SECONDARY
  HOST: blue
  JOURNAL:  /var/opt/SFTKdtc
#
#
#  Device Definitions:
#
PROFILE:
   REMARK:  /opt/www Filesystem
   PRIMARY:SYSTEM-A
   DTC-DEVICE:/dev/dtc/lg0/rdsk/dtc0
   DATA-DISK:
   SECONDARY:SYSTEM-B
   MIRROR-DISK:
#
#
# -- End of dtc Configuration File:   /etc/opt/
SFTKdtc/p000.cfg
```

B

Throttle Reference

Throttle Format 137

Measurement Keywords 138

Actions 139

Throttle Format

The general throttle format for Softek Replicator 2.1 is shown below:

```
THROTTLE[:] <dow-dom> <from-time> <to-time> <throttle-tests>
        ACTIONLIST[:]
            <actions>
        ENDACTIONLIST[:]
```

The colon is optional.

You can break up long lines in throttles into multiple, more readable lines by placing a "\" character by itself at the end of a line that is continued on the next line.

For example:

```
THROTTLE: - - - PCTCPU > 30 AND \
ACTUALKBPS > 1000
```

The throttle fields are as follows:

`<dow-dom>` = "-" means no day-of-week or day-of-month sensitivity

or

`<dow-dom>` = one or more comma-separated items (no spaces allowed) consisting of days of the month or two-character days of the week mnemonics (for example, mo,tu,we,th,fr,sa,su), or an "em" mnemonic, which always maps to the last day of the current month, or any or all of these in no specific order. These mnemonic values are case-insensitive and duplicates are allowed. For example:

```
mo,tu,we,th,fr
1,15,em
1,mo,EM,15,4,15,SU,we
```

`<from-time>` = "-" no time sensitivity for this throttle

or

`<from-time>` = a time specified in HH:MM:SS format after which the throttle may be evaluated. For example:

```
01:00:00
12:59:00
18:15:00
```

`<to-time>` = "-" no time sensitivity for this throttle

or

`<to-time>` = a time specified in HH:MM:SS format before which the throttle may be evaluated. `<to-time>` must be later than `<from-time>`. Examples:

```
01:00:10
12:59:59
22:45:00
```

`<throttle-tests>` = one or more test clauses expressed in terms of a measurement name, a relational operator, and a value. If more than one test clause is given, a logical operator of either "AND" or "OR" must be specified between the test clauses.

If these test clauses evaluate to TRUE, the subsequent ACTIONLIST is executed. If the test clause evaluate to FALSE, then the subsequent ACTIONLIST is not executed.

The test clauses are evaluated from left to right. Up to 16 test clauses linked by logical operators in a throttle test are allowed. The measurement keywords are case insensitive.

For example:

```
EFFECTKBPS T> 500
DRIVERMODE == 1 AND NETCONNECTFLAG != 1
```

Measurement Keywords

The following are the supported throttle measurement keywords:

NETKBPS -- network throughput in kilobytes per second. This is the same as EFFECTKBPS.

PCTCPU -- percentage of system CPU that the PMD for this logical group is using.

PCTBAB -- percentage of the BAB that this logical group has in use. This is the same as PERCENTBABINUSE.

NETCONNECTFLAG -- set to "1" if the PMD is connected to the RMD for this logical group, "0" otherwise.

PID -- process id (as returned by the **ps** command) for the PMD for this logical group (or "-1" if PMD is not running).

CHUNKSIZE -- maximum size of a single data packet that the PMD sends to the RMD. the default value is 2048KB, with 64KB as the minimum and 16384KB as the maximum.

STATINTERVAL -- number of seconds between each performance update calculation and throttle evaluation. The minimum and maximum values are 1 and 86400, respectively, with the default set to 10 seconds.

MAXSTATFILESIZE -- number of kilobytes that a performance file is allowed to grow to. The minimum and maximum values are 1 and 32000, respectively, with the default set to 64KB.

STATTOFILEFLAG -- if set to "1" (on), performance information is added to the performance files. If set to "0" (off), the performance files are not updated. The default is "1".

TRACETHROTTLE -- if set to "1" (on), every throttle that evaluates to TRUE and the corresponding ACTION executed is logged to syslog and the Softek Replicator error log. If set to "0" (off), these are not logged. The default is "0".

SYNCMODE -- if set to "0" (off), this logical group is in asynchronous transfer mode. If set to "1" (on), this logical group is in synchronous, or semi-synchronous transfer mode. The default is "0".

SYNCMODEDEPTH -- the depth of I/O updates that may accumulate in Sync mode before the dtc device driver will block the application. If this is set to “1”, then this logical group is in Sync mode. If this value is greater than “1”, then this logical group is in semi-synchronous mode. The minimum and maximum values are 1 and 2147483647, respectively, with the default set to 1.

SYNCMODETIMEOUT -- the number of seconds that the dtc device driver will allow a synchronous or semi-synchronous update to be sent to the secondary system, and wait for an acknowledgment of committed update by the RMD before moving on to the next update pending. The minimum and maximum values are 1 and 86400, respectively, with the default set to 30 seconds.

COMPRESSION -- if set to "1" (on), moderately effective compression is applied to updates sent by the PMD to the RMD. If set to "0" (off), only trivial zero-filled block discard compression is employed. The default is "0".

NETMAXKBPS -- limit imposed on the PMD as to how many kilobytes per second (on average) will be allowed to occupy the network bandwidth between the primary and secondary systems. The minimum and maximum values are 1 and 2147483647, respectively, with the default set to -1 (off).

CHUNKDELAY -- the number of milliseconds the PMD sleeps after sending a packet of data to the RMD. This is a tunable parameter for moderating CPU utilization and is separate from the NETMAXKBPS regulating mechanisms in the PMD, but affects the network throughput realized. The minimum and maximum values are 0 and 999, respectively, with the default set to 0 milliseconds.

DRIVERMODE -- may assume one of the following values:

0 = NORMAL mode -- coherent update collection/transmission

1 = TRACKING mode -- collecting changed block pointers for a smart REFRESH

2 = PASSTHRU mode -- all operations directed to local data devices only

3 = REFRESH mode -- performing a refresh on TRACKING mode data and collecting new updates coherently

4 = BACKFRESH mode -- updating the local data devices from the mirror devices on the secondary

ACTUALKBPS -- the number of actual physical kilobytes sent from the PMD to the RMD per second over the network.

EFFECTKBPS -- the effective kilobytes per second sent by the PMD to the RMD over the network (decompressed).

PERCENTDONE -- the percentage that a refresh or backfresh operation has completed for a logical group.

ENTRIES -- the number of I/O updates received by the dtc device driver awaiting transmission to the secondary system.

PERCENTBABINUSE -- the percentage of occupation of the BAB by entries awaiting transmission to the secondary system.

Actions

Specify throttle actions with the following keyword:

`<action>` = a directive to do something with optional values or messages.

You can include up to 16 actions in an ACTIONLIST.

Action Directives

The following is a list of possible action directives you can use in throttles:

ACTION[:] do console `<message>`

send a message to the primary system's console

ACTION[:] do mail `<to-whom> <message>`

send mail message

ACTION[:] do exec `<program-or-script> <arguments>`

execute an external program or script

ACTION[:] do log `<message>`

log a message in the syslog and in the dtcerror.log file

ACTION[:] set `<tunable> <value>`

set a tunable parameter to a specific value

ACTION[:] incr `<tunable> <amount-value>`

increment a tunable parameter value by an amount

ACTION[:] decr `<tunable> <amount-value>`

decrement a tunable parameter value by an amount

Tunable Action Parameters

You can set the following tunable parameters using a throttle:

- CHUNKDELAY
- CHUNKSIZE
- STATINTERVAL
- MAXSTATFILESIZE
- TRACETHROTTLE
- Sync modeSYNCMODE
- Sync modeSYNCMODEDEPTH
- Sync modeSYNCMODETIMEOUT
- COMPRESSION

Action Message Substitutions

Action message substitutions must be in uppercase within ACTION statements and must be specified with a leading "%%" and trailing "%%" . They must not be imbedded in another word. An example of an action message substitution is: "do mail root effective kbps = %%EFFECTKBPS%%".

PID	PMDs process id (as from `ps` command).
GROUPNO	Current logical group number.
CFGFILE	Path to the configuration file for logical group.
CPU	CPU percentage consumed by PMD process.
KBPS	Effective kilobytes per second transmitted.
PCTWL	Percent of the BAB currently in use.

SLEEP	Current setting for CHUNKDELAY.
TIME	Current time as "hh:mm:ss".
DATE	Current date as "mm dd, yyyy".
NETCONNECTFLAG	1=PMD is actively in communication with RMD.
CHUNKSIZE	Maximum size of packets sent to secondary system.
STATINTERVAL	Number of seconds between throttle evaluations.
MAXSTATFILESIZE	Maximum size performance files will grow in KBs.
STATTOFILEFLAG	1=log performance information, 0=do not log performance information.
TRACETHROTTLE	Write message to *syslog/dtcerror.log* for executing actions.
SYNCMODE	1=sync or semi-sync mode, 0=async mode.
SYNCMODEDEPTH	1=sync mode, >1=semi-sync mode maximum growth.
SYNCMODETIMEOUT	Seconds in which sync mode is honored for each update sent.
COMPRESSION	1=best compression employed, 0=trivial compression employed.
NETMAXKBPS	Maximum kilobytes per second (average) allowed over net.
CHUNKDELAY	Milliseconds slept between packets sent over net.
DRIVERMODE	NORMAL, TRACKING, PASSTHRU, REFRESH, or BACKFRESH.
ACTUALKBPS	Physical kilobytes sent over the network per second.
EFFECTKBPS	Decompressed kilobytes received by the RMD.
PERCENTDONE	Percentage the refresh or backfresh has completed.
PERCENTBABINUSE	Percentage of BAB with data awaiting transfer.

C

Tunable Parameters

CHUNKDELAY 145

CHUNKSIZE 145

COMPRESSION 145

MAXSTATFILESIZE 145

NETMAXKBPS 146

STATINTERVAL 146

SYNCMODE 146

SYNCMODEDEPTH 147

SYNCMODETIMEOUT 147

TRACETHROTTLE 147

This appendix provides a listing of all tunable parameters and a description of their functionality.

Note that you must run `killpmds` before and `launchpmds` after changing the following tunable parameters:

- CHUNKSIZE
- COMPRESSION
- NETMAXKBPS

Note that you must run `dtcstop` before and `dtcstart` after changing the following tunable parameters:

- SYNCMODE
- SYNCMODEDEPTH
- SYNCMODETIMEOUT

CHUNKDELAY

This parameter sets the amount of time, in milliseconds, that the PMD waits before attempting to send a chunk of entries from the BAB across the network.

CHUNKDELAY is simply a method to regulate the performance of a PMD. Consider using this parameter to regulate the amount of network bandwidth consumed by Softek Replicator. Setting CHUNKDELAY to a value greater than 0 causes the PMD to wait the designated amount of time before attempting to transfer data. The default is 0.

CHUNKSIZE

This parameter sets the amount of information, in KBytes, that the PMD reads from the BAB at a time and sends across to the RMD.

The CHUNKSIZE parameter can help optimize disk access while Softek Replicator is performing a refresh. Each operation reads the amount of data defined by CHUNKSIZE. The default is 2048KB.

Be aware when setting this parameter, that an I/O operation that reads or writes a data amount greater than the value of CHUNKSIZE may cause the BAB to overflow.

COMPRESSION

This parameter sets data compression. The default is OFF, which indicates that data being transferred is in its original, non-compressed state. If set to ON, then data is compressed before being transferred and decompressed before being written to the journal or mirror device on the secondary system. Compression is a means to optimize network throughput.

MAXSTATFILESIZE

This parameter governs the size, in kilobytes, for the performance file when LOGSTATS is set to “Y”. Once a performance file reaches the size set by this parameter, the file is closed and renamed with a .1 suffix. Only one generation of files is saved. Then the original file is emptied and readied to collect new performance metrics.

This parameter allows you to establish the amount of disk space for these performance statistics. The frequency at which these files are updated is determined by STATINTERVAL (see *STATINTERVAL* on page 146), so in a round about way, MAXSTATFILESIZE also allows you to save a larger number of statistics, spread out over a long period of time (by increasing the value of this parameter)— if the goal is to decipher trends over a long operating period. Or the operator can choose to limit the statistics (by reducing the value of this parameter) to only a short period of time. The default is 64 KB.

NETMAXKBPS

This setting regulates the network bandwidth, in kilobytes per second, Softek Replicator uses. The following figure shows an example of how this tunable parameter can be set up in a throttle to control network bandwidth during specific hours. The default is -1 (OFF).

Example of Network Bandwidth Throttle

```
THROTTLE: MO,TU,WE,TH,FR 08:00:00 17:00:00 \
NETMAXKBPS != 300
  ACTIONLIST:
  ACTION: set NETMAXKBPS 300
  ENDACTIONLIST:
THROTTLE: MO,TU,WE,TH,FR 17:01:00 23:59:59 \
NETMAXKBPS != 1000000
  ACTIONLIST:
  ACTION: set NETMAXKBPS 1000000
  ENDACTIONLIST:
THROTTLE: MO,TU,WE,TH,FR 00:00:00 07:59:59 \
NETMAXKBPS != 1000000
  ACTIONLIST:
  ACTION: set NETMAXKBPS 1000000
  ENDACTIONLIST:
THROTTLE: SA,SU - - NETMAXKBPS != 1000000
  ACTIONLIST:
  ACTION: set NETMAXKBPS 1000000
  ENDACTIONLIST:
```

STATINTERVAL

This parameter defines the frequency, in seconds, at which Softek Replicator evaluates throttles, tunable parameters and performance metrics. The default is 10 seconds.

SYNCMODE

This parameter determines whether the dtc devices in a logical group require a synchronous and acknowledged update from the secondary mirror device with each I/O update to the local data device. When set to 1 (on), all I/O operations are performed on both the local data and mirror devices disks at all times. Setting this parameter to 1 (on) affects application performance because of round trip network time added to each I/O. Setting this parameter to 0 (off) allows transfers to be made asynchronously. The default is 0 (off).

SYNCMODEDEPTH

This option sets the number of I/Os that can accumulate in the BAB before Sync mode is triggered. If SYNCMODEDEPTH is set to 1 and SYNCMODE is set to Y, then dtc devices in the logical group are in full Sync mode. If SYNCMODEDEPTH is set to a value larger than 1, then the logical group is in Async mode, but the mirror devices on the secondary are no more than the specified number of entries (set by this parameter) behind the local data devices. The default is 1.

SYNCMODETIMEOUT

This parameter establishes the amount of time, in seconds, that the dtc device driver waits for a Sync mode update to complete before returning control to the application, just as if the acknowledgment of the mirror device update had been received. If the PMD does not receive an acknowledgment, the status of its logical group changes to TRACKING mode automatically. This does not indicate that Softek Replicator has switched to Async mode, but rather this parameter keeps the application from freezing up if there is a high load burst on either the network or the secondary. The default is 30 seconds.

TRACETHROTTLE

This parameter determines whether throttle information is written to the `syslog` (`/var/adm/messages` in AIX and Solaris, `/var/adm/syslog/syslog.log` in HP-UX, and `/var/log/messages` in Linux) and to the error log file (`/var/opt/SFTKdtc/dtcerror.log`) when a throttle evaluates TRUE. If this parameter is set to either “on”, “ON”, or “1”, then each ACTION in the ACTIONLIST for the throttle is written out to these files along with the current values of variables included in the throttle definition. This provides the operator with a method of verifying that a throttle executed properly. The default is OFF.

NOTE **For AIX only**
The path to the error log file is, as follows:
`/var/dtc/dtcerror.log`

■ ■

D

Sample Script

Using the Sample Script 151

This appendix describes the sample scripts that can be used for changing Symbolic link described in Step 6 of *Configuring Softek Replicator for Relational Database Management Systems (RDBMS)* on page 53.

The file names of the sample scripts are as follows:

- AIX sample script

  `/usr/dtc/samples/chlink`

- HP-UX sample script

  `/opt/STFKdtc/samples/chlink`

- Linux sample script

  `/opt/SFTKdtc/samples/chlink`

- Solaris sample script

  `/opt/STFKdtc/samples/chlink`

Using the Sample Script

To use the sample script, proceed as follows:

1) Make a copy of the sample script and study it.
2) Modify the script to meet your settings.
3) Change the execution permission and run the script.

Options

Option	Comment
-g <group#>	Logical group number
SYMBOLIC_LINK_name	Symbolic link name that is targeted to change, in absolute path form.

Sample input:

```
myserver# chlink -g 3 /oracle/space3
```

Output:

The following message is displayed:

```
**Warning** This script needs to be modified to meet your Oracle
settings, and please run this script on your own responsibility.
Change Symbolic link /oracle/space3
Original Link destination is: /dev/rdsk/c0t1d3s1
New Link destination is: /dev/dtc/lg3/rdsk/dtc1
Do you want to change it now? (Y or N) > Y
Completed
```

CAUTION:
Before running the sample script, you must perform Steps **1** through **5**, under *Configuring Softek Replicator for Relational Database Management Systems (RDBMS)* on page 53.

If you use this script for Oracle, the Oracle instance which is targeted to change, must be stopped.

You must provide a valid definition for `SYMBOLIC_LINK_name` and `-g <group#>` (logical group).

The sample script is provided as is and is not guaranteed to function in all environments. You must modify the script to reflect your specific environment. If you run the sample script without modifying it, you will encounter errors.

The sample script changes the specified symbolic link, for one dtc device at a time. If the specified logical group includes multiple dtc devices, you must run the script multiple times against the logical group.

Uninstalling Softek Replicator will delete the original sample script.

Glossary

NOTE This glossary contains terms and definitions found in this manual. Most of the terms are specific to Softek Replicator.

A

Accumulate
The Softek Replicator mode in which the PMD and the RMD are not operating for the logical group, and the updates to Softek Replicator devices are accumulating in the BAB.

Action
A directive in a throttle definition to do something with optional values and messages.

ACTIONLIST
The component of a throttle which lists the *actions* defined for that throttle.

Actual KBps
The actual amount of data, in KBps, sent over the network reported by **`dtcperftool`**.

Aggregated Throughput
A group of network connections whose throughput is considered as a whole.

Automatic Recovery
Softek Replicator's action of going into Tracking Mode and then launching a smart refresh when the BAB exceeds its limits.

B

BAB
The buffer cache in physical memory used to store transactions waiting to be mirrored to the secondary system.

Backfresh Mode
The maintenance mode in which data is being copied from the current secondary system to the current primary system.

C

.cfg Files
Logical group configuration files, created by `dtcconfigtool` during Softek Replicator configuration. These files are processed by `dtcstart`. All .cfg files need to be copied over the secondary system and renamed.

Chaining
A configuration option in which systems appear to be connected in a sequential manner, such as A to B to C.

Checkpoint Scripts
Optionally implemented scripts for automating pre- and post- checkpoint tasks.

Checkpointing
The act of taking a snapshot image of data on the mirror devices and making this data accessible in a read-only mode to applications other than Softek Replicator.

CHUNKDELAY
A parameter that defines, in milliseconds, determined by the operator, which designates the length of time the PMD is idle between transfers of data.

CHUNKSIZE
A parameter that defines the amount of data, in KBytes, that is sent across the network during a transfer.

.cur Files
These are copies of the **`.cfg`** (logical group configuration files), which `dtcstart` creates when it processes a **`.cfg`** file. These files are accessed and referenced by all other Softek Replicator commands during operations. The dtcstop command deletes the **`.cur`** file for specified logical groups.

D

Data Coherence
The integrity of information written to dtc devices.

Data Migration
The periodic passing of data between primary and secondary systems.

Dynamic Allocation
The *as needed* method of distribution of the memory assigned to the BAB between logical groups.

E

Effective KBps
The amount of data being transferred across the network without compression.

F

File System
Abstract data types that are necessary for the storage, hierarchical organization, manipulation, navigation, access and retrieval of data in modern operating systems.

Most operating systems provide a file system.. File systems are represented either textually or graphically by file browsers or shells. If graphically, the metaphor of *folders* containing documents, other files, and nested folders is often used.

Full Refresh
The process in which each block of data on the local data device is copied to the mirror device.

I

in.dtc
The master Softek Replicator daemon process running on all primary and secondary systems in the company, which is responsible for launching and managing all other processes.

Incoherent State
The data that is lacking continuity or is in an inconsistent state. In the context of Softek Replicator, this is a temporary state from which data can be recovered.

Initial Mirror
The untouched copy of the original data set on the secondary system.

J

Journal File Directory
The user-defined directory or directories on the secondary system where journal files are written.

Journaling
The act of recording current updates in a log file (journal) on the secondary system rather than writing the data directly to the mirror device.

L

License Key
A 24 character alphanumeric string placed in the license file (*DTC.lic*) that unlocks the daemon processes and allows Softek Replicator to be operational.

Local Data Device
A disk partition or managed volume on a primary system that stores all or a portion of the original data set. A local data device is the primary system side of the dtc device.

Logical Group
A virtual, not physical, grouping of dtc devices into a cohesive unit across which data coherence is maintained.

LOGSTATS
A tunable parameter indicating whether performance statistics should be logged in the performance files. The default is "Y".

M

MAXSTATFILESIZE
A tunable parameter that designates the maximum size (in KBytes) for the file of performance metrics, if statics are being logged. The default is 64 KB.

Mirror Device
A device configured on the secondary system, on which an exact replica of the original data set is stored.

N

NETMAXKBPS
A tunable parameter that designates the maximum amount of data, in KBytes per second, to be transferred across the network. This is a means of controlling how much bandwidth Softek Replicator consumes.

Normal Mode
A healthy operational state in which all typical network data mirroring elements are functional.

O

.off File
A file placed in the /etc/opt directory by the `dtcrmdreco` command that prohibits mirroring to the secondary mirror devices. Typing a `dtcrmdreco -d` command deletes this file.

Original Data Set
The data set residing on the primary system.

P

p###.cfg Files
Logical group configuration files created by `dtcconfigtool` during configuration. These files are processed by `dtcstart`.

p###.cur Files
Copies of the .cfg files created by `dtcstart`. These files are referenced by all Softek Replicator commands during normal operations.

Passthru Mode
A transitional mode of operation in which data is not being mirrored to the secondary system. Softek Replicator operates in this mode initially after installation.

PMD
The primary mirror daemon (PMD) is a background process on the primary system responsible for establishing a connection with the secondary system, and transferring data from the BAB to the appropriate mirror device.

Primary System
System(s) in Softek Replicator in which the original data set resides, local data devices are configured and applications are active.

Pstore
The persistent storage (pstore) is allocated during configuration for state information and contains tunable parameter definitions.

R

Recoverable State
The condition of data at a point in time when it cannot be accessed and used without re-establishing the usability of the data. For example, data on the mirror devices is in a *recoverable state* during a refresh operation, or while entries are being copied from the journal files to the mirror devices.

Refresh Mode
An operational mode in which the data on the mirror devices is being renewed from the local data devices.

RMD
The remote mirror daemon (RMD) is the background process on the secondary system that handles the writing of data to the mirror devices or to the journals.

S

Secondary System
A system in Softek Replicator that stores a mirror copy of the original (primary) data set to be accessed during planned or unplanned outages for the purpose of disaster recovery or maintenance.

Smart Refresh
The process of transferring only blocks of data that have recently changed from the local data device to the mirror device.

s###.off Files
Files created by the `dtcrmdreco` command that prohibit mirroring to the mirror devices on the secondary system.

STATINTERVAL
A parameter that defines the time, in seconds, that elapses before the throttle process calculates performance metrics and evaluates all throttles. The default is 10 seconds.

Synchronization
The process that ensures that all systems in the company are loaded with identical sets of contemporary data.

SYNCMODE
A tunable parameter that governs whether a synchronous and acknowledged update is required from each secondary system with every I/O.

SYNCMODEDEPTH
A tunable parameter that determines the number of entries allowed to accumulate asynchronously in the BAB before being transferred and acknowledged by the secondary system. If this value is set to a number greater than 1, the dtc devices are t

throtd
The Softek Replicator daemon process responsible for evaluating and executing throttles.

Throttle
An optional user-defined gauge that can fine-tune certain Softek Replicator functions by monitoring elements, measuring variables, and performing actions.

Throttle Editor
The function in `dtcconfigtool` where you define or edit throttle definitions.

Time Sequenced Transfers
A method through which Softek Replicator ensures data coherence by preserving the order of consecutive updates to the local data device as they are written to the BAB and later transferred to the mirror device.

TRACETHROTTLE
A tunable parameter that monitors the execution of throttles and writes each ACTION and current variable values to the syslog and the error log file. The default is "OFF".

Tracking Mode
The operational mode in which updates to the local data device are not written to the BAB, but are rather recorded in a disk map.

Tunable Parameters
The optional user-defined variables, stored in the pstore, that allow fine-tuning of certain Softek Replicator components.

Typical Operations
Tasks representative of Softek Replicator network data mirroring, such as writing simultaneously to the BAB and the local data device, and reading entries from the BAB and sending them to the appropriate mirror device.

U

Unplanned Outage
An inadvertent loss of one or more components in the data mirroring enterprise that disrupt normal activity.

Index

Symbols

.cfg files
copying to secondary 22, 31, 39, 47
distributing 22, 30, 39, 47
example 133
overview 22, 30, 39, 47
renaming 31, 39, 47
throttle definitions in 59
used by start command 111
.cur files 23, 31, 39, 47, 111
.off files 109
/etc/vfstab
modifying the file 32, 48

A

actual kbps 88, 139
applications
product 4
quieting before checkpoint 74
Async mode 6, 12, 147

B

BAB
defining 35
overflow during refresh 69
overview 5, 9
recovering from overflow 83
resizing 68
resizing manually 69
tracking overflow 92
Backfresh Mode 15
Backfresh operation
after a recovery 83
command 115
overview 15
backup 4
block device driver 8

C

chaining
defining 21, 29, 37, 45
example 127
checkpointing
activating 70
considerations and limitations 76
file systems 74
Normal Mode 76
primary system scripts 72
secondary system scripts 73
shell scripts 71
stopping 71
token in BAB 74
checksum refresh 15, 108, 117
chunkdelay 139, 140, 145
chunksize 51, 138, 140, 145
coherency
using multiple logical groups 10
coherent 12
complex configurations
loopback configuration 129
symmetic configuration 121
compression 51, 92, 139, 140, 145
configuration
chaining 127
current 23, 31, 39, 47
files 22, 31, 39, 47
Linux 35
logical group 22, 30, 39, 47
loopback 129
one to many 125
symmetric configuration 121
courier transport method 26, 34, 42, 50

D

daemons
primary 5, 7
remote 11
data flow
monitoring 92
data migration 4
data recovery 79
databases
configuring product with 53
non-symbolic linked devices 55
DBMS

tablespace allocation 11
defining
logical group (Linux) 36
device status area 90, 91
devices
block 8
failure 84
local data 10
mirror 22, 30, 38, 46
special character 8
disaster recovery 81
overview 4
steps 81
distributing
configuration files 39
dom 137
dow 137
drivermode 139
dtc device
and logical groups 5
dtcautomount 97
dtcoverride 41
dtcperftool 107

E

effectkbps 88, 138, 139
entries 139
entry age recvd 89

F

File Systems
Configuring AIX 24
Configuring HP-UX 32
Configuring Linux 40
fsck command
for checkpointing 70
in data recovery 82
full refresh 15, 117

I

incoherent 12

J

j###.###.c 12
j###.###.i 12
journal file directory
adding to /etc/vfstab 32, 48
journal files 12

K

kernel buffer 9
killbackfresh 97
killpmds 113
killrefresh 114
killrmds 115

L

launchbackfresh 115
launchpmds 116
launchrefresh 41, 117
Linux
configuration 35
local data device 5
logical group 9
configuration file example 133
defining 20, 28, 44
defining for Linux 36
defining pstores for 20, 28, 36, 44
deleting 67
logstats 145
loopback configuration 20, 28, 36, 44, 129

M

mapping
local and mirror devices 10
master daemon port numbers 38
maxstatfilesize 51, 90, 138, 140, 145, 146
mdsset 110
mincache=dsync 33, 49
mirror device
defining 22, 30, 38, 46
mirroring 3, 25, 33, 41, 49
monitoring
BAB 92
data flow 92
performance files 89
tunable parameters 90
mount command 33, 49, 82
mountall command 33, 49

N

Near sync mode 12
netconnectflag 138
netkbps 138

netmaxkbps 51, 139, 146
network mirroring 3
networks 10
Normal Mode
transitioning to 25, 34, 41, 50

O

one to many 125
operating modes
Backfresh Mode 15
monitoring 13
Normal mode 14
Passthru Mode 13
Refresh Mode 14
Tracking Mode 14

P

p###.cfg 22, 30, 39, 47
p###.off 109
p###.prf 89
Passthru Mode 13, 24, 32, 40, 48, 122
pctbab 138
percentbabinuse 139
percentdone 139
persistent store 9
pid 138
PMD 5, 7, 145
port number 38
changing connecting port 21, 28, 36, 44
primary system
components 7
failure 82
restoring after failover 83
pstore
defining for logical group 20, 28, 36, 44
relocating 68
sizing 9
state information 9

R

rdbms environments 53
recovery 79
Refresh Mode 14
refresh operation
after a recovery 84
backfresh 115
checksum 15
command 117
full 15
launchrefresh 25, 34, 41, 50
RMD 11

S

secondary
components 11
defining address 20, 28, 36, 44
failing over to 83
journaling 12
sizing
pstore 9
smart refresh 15, 108
statinterval 51, 90, 138, 140, 146
stattofileflag 138
status message area 91
symmetric configuration 121
Sync mode 12, 146, 147
synchronizing
after a failure 82
after installation 24, 32, 40, 48
system maintenance 4

T

throttles
action directives 139
action messages 140
actions 61, 139
defining 22, 30, 38, 46, 60
editing 59, 81
examples 62
expressions 60
format 137
keywords 138
overview 12
setting evaluation time 60
tests 137
throttle builder 60
tracethrottle 51, 138, 140, 147
Tracking Mode 14
tunable parameters 38
defining 51, 63
editing 62, 63
overview 12
valid values 145

U

umount command 33, 49

Unix operating systems supported 3

Revision History

This revision history lists all revisions of this publication and their effective dates.

Revision Level	Change Summary
ML-145086-001 December 2003	General Availability release of this manual for Softek Replicator, Version 2.1

SOFTEK
Simplifying Storage Management

Softek Offices

Worldwide Headquarters – USA
1250 East Arques Avenue, M/S 317
Sunnyvale, CA 94085
Toll–free 1.877.887.4562
Phone 408.746.7638
Facsimile 408.737.5900
Email USA cs@softek.fujitsu.com

United Kingdom
1 Meadow Gate Avenue,
Farnborough Business Park
Farnborough, Hampshire GU14 6FG
Phone 44.1252.550440
Facsimile 44.1252.550441
Email Europe info@softek.fujitsu.com

Canada
Softek Software Technology, Ltd.
Phone 450.686.2455
Facsimile 450.686.0239

Ireland
Softek Software Technology Ireland, Ltd.
Phone 353.1.813.6000
Facsimile 353.1.813.6321

France, Spain & Portugal
Softek Software Technology SAS
Phone 33.1.44.10.41.00
Facsimile 33.1.44.10.41.08

Italy
Softek Software Technology, Filiale Italiana
Phone 39.02.241.01405
Facsimile 39.02.241.01408

Germany, Austria, Switzerland
Softek Software Technology, GmbH
Phone 49.89.244417.0
Facsimile 49.89.244417.111

Belgium, Luxembourg
Softek Software Technology
Phone 32.2.712.77.72
Facsimile 32.2.712.78.19

The Netherlands
Softek Software Technology
Phone 31.346.244700
Facsimile 31.346.244710

For more information, please visit us at www.softek.com