HP Serviceguard Extended Distance Cluster for Linux A.01.00 Deployment Guide

Manufacturing Part Number: T2808-90006

May 2008 Second Edition

Legal Notices

Publication Date: 2008

Contents

Contents

Contents

Contents

Printing History

Table 1 Editions and Releases

Printing Date	Part Number	Edition	Operating System Releases (see Note below)
November 2006	T2808-90001	Edition 1	• Red Hat 4 U3 or later • Novell SUSE Linux Enterprise Server 9 SP3 or later • Novell SUSE Linux Enterprise Server 10 or later
May 2008	T2808-90006	Edition 2	• Red Hat 4 U3 or later • Novell SUSE Linux Enterprise Server 9 SP3 or later • Novell SUSE Linux Enterprise Server 10 or later

The printing date and part number indicate the current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The part number changes when extensive technical changes are incorporated. New editions of this manual will incorporate all material updated since the previous edition.

NOTE This document describes a group of separate software products that are released independently of one another. Not all products described in this document are necessarily supported on all the same operating system releases. Consult your product's *Release Notes* for information about supported platforms.

HP Printing Division:

Business Critical Computing Business Unit
Hewlett-Packard Co.
19111 Pruneridge Ave.
Cupertino, CA 95014

Preface

This guide introduces the concept of Extended Distance Clusters (XDC). It describes how to configure and manage HP Serviceguard Extended Distance Clusters for Linux and the associated Software RAID functionality.

In addition, this guide includes information on a variety of Hewlett-Packard (HP) high availability cluster technologies that provide disaster tolerance for your mission-critical applications. Serviceguard has supported disaster tolerant clusters on HP-UX for several years now while it is relatively new on Linux. Features of those disaster tolerant HP-UX systems may be used as an example through this document.

Intended Audience

It is assumed that you are familiar with the following topics

- HP Serviceguard configurations
- Basic RAID concepts

Document Organization

The chapters of this guide include:

- Chapter 1, “Disaster Tolerance and Recovery in a Serviceguard Cluster,” on page 13.
- Chapter 2, “Building an Extended Distance Cluster Using Serviceguard and Software RAID,” on page 51
- Chapter 3, “Configuring your Environment for Software RAID,” on page 61
- Chapter 4, “Disaster Scenarios and Their Handling,” on page 85

Preface

Related Publications

The following documents contain additional useful information:

- *Clusters for High Availability: a Primer of HP Solutions, Second Edition*. Hewlett-Packard Professional Books: Prentice Hall PTR, 2001 (ISBN 0-13-089355-2)
- *Designing Disaster Tolerant HA Clusters Using Metrocluster and Continentalclusters* (B7660-900xx)
- *HP StorageWorks Cluster Extension EVA user guide*
- *HP StorageWorks Cluster Extension XP for HP Serviceguard for Linux*
- *HP Serviceguard for Linux Version A.11.16 Release Notes*
- *Managing HP Serviceguard for Linux*

Use the following URL to access HP's High Availability web page:

- **`http://www.hp.com/go/ha`**

Problem Reporting

If you have problems with HP software or hardware products, please contact your HP support representative.

1 Disaster Tolerance and Recovery in a Serviceguard Cluster

This chapter introduces a variety of Hewlett-Packard high availability cluster technologies that provide disaster tolerance for your mission-critical applications. It is assumed that you are already familiar with Serviceguard high availability concepts and configurations.

This chapter covers the following topics:

Evaluating the Need for Disaster Tolerance

Disaster tolerance is the ability to restore applications and data within a reasonable period of time after a disaster. Most people think of fire, flood, and earthquake as disasters, but a **disaster** can be any event that unexpectedly interrupts service or corrupts data in an entire data center: the backhoe that digs too deep and severs a network connection, or an act of sabotage.

Disaster tolerant architectures protect against unplanned down time due to disasters by geographically distributing the nodes in a cluster so that a disaster at one site does not disable the entire cluster. To evaluate your need for a disaster tolerant solution, you need to weigh:

- *Risk of disaster.* Areas prone to tornadoes, floods, or earthquakes may require a disaster recovery solution. Some industries need to consider risks other than natural disasters or accidents, such as terrorist activity or sabotage.

  The type of disaster to which your business is prone, whether it is due to geographical location or the nature of the business, will determine the type of disaster recovery you choose. For example, if you live in a region prone to big earthquakes, you are not likely to put your alternate or backup nodes in the same city as your primary nodes, because that sort of disaster affects a large area.

  The frequency of the disaster also plays an important role in determining whether to invest in a rapid disaster recovery solution. For example, you would be more likely to protect from hurricanes that occur seasonally, rather than protecting from a dormant volcano.

- *Vulnerability of the business.* How long can your business afford to be down? Some parts of a business may be able to endure a 1 or 2 day recovery time, while others need to recover in a matter of minutes. Some parts of a business only need local protection from single outages, such a node failure. Other parts of a business may need both local protection and protection in case of site failure.

  It is important to consider the role applications play in your business. For example, you may target the assembly line production servers as most in need of quick recovery. But if the most likely disaster in your area is an earthquake, it would render the assembly

line inoperable as well as the computers. In this case disaster recovery would be moot, and local failover is probably the more appropriate level of protection.

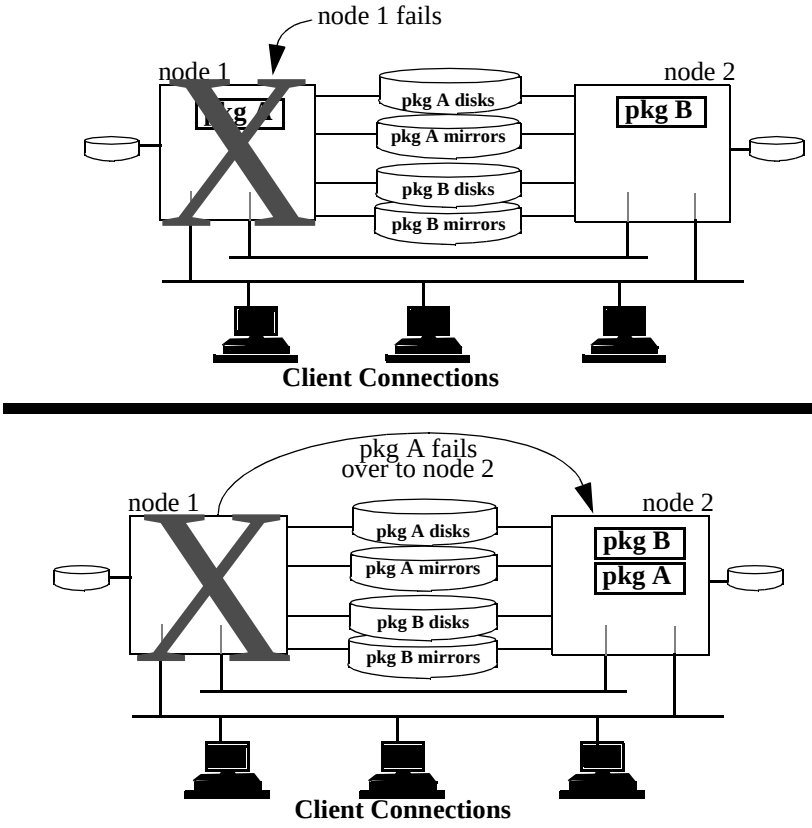
On the other hand, you may have an order processing center that is prone to floods in the winter. The business loses thousands of dollars a minute while the order processing servers are down. A disaster tolerant architecture is appropriate protection in this situation.

Deciding to implement a disaster recovery solution really depends on the balance between risk of disaster, and the vulnerability of your business if a disaster occurs. The following pages give a high-level view of a variety of disaster tolerant solutions and sketch the general guidelines that you must follow in developing a disaster tolerant computing environment.

What is a Disaster Tolerant Architecture?

In a Serviceguard cluster configuration, high availability is achieved by using redundant hardware to eliminate single points of failure. This protects the cluster against hardware faults, such as the node failure in Figure 1-1.

Figure 1-1 High Availability Architecture.

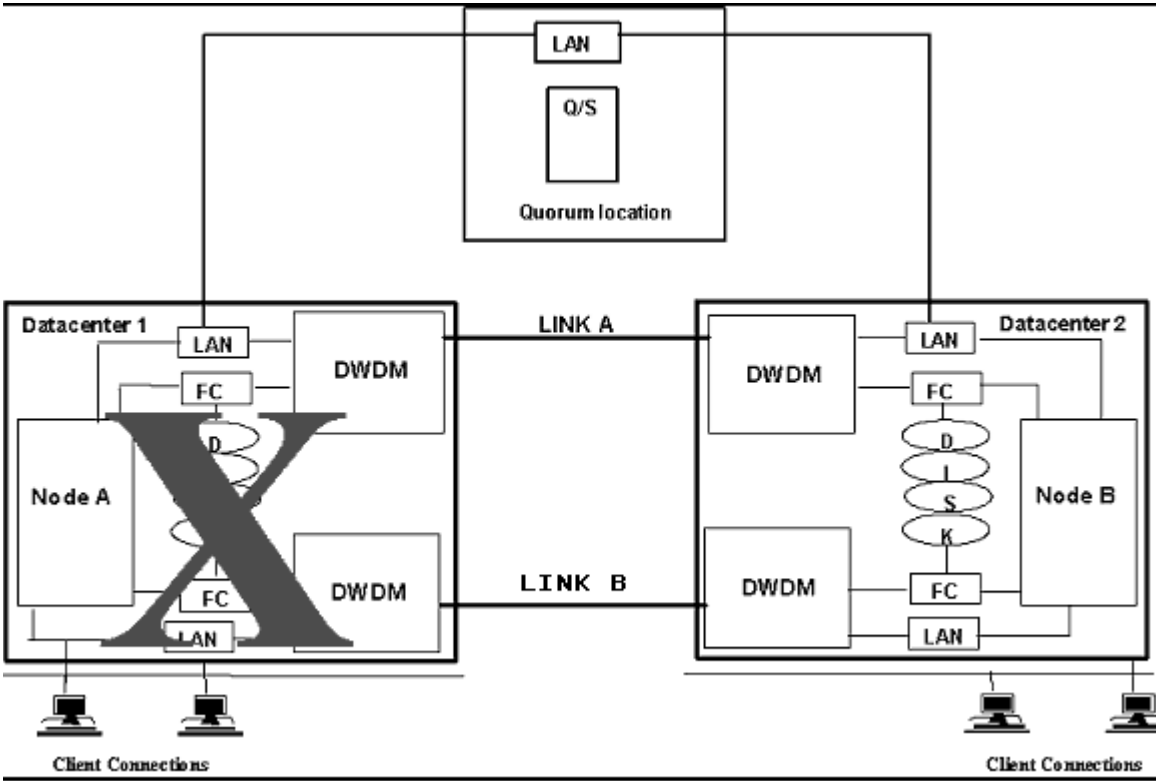


This architecture, which is typically implemented on one site in a single data center, is sometimes called a **local cluster**. For some installations, the level of protection given by a local cluster is insufficient. Consider the order processing center where power outages are common during harsh weather. Or consider the systems running the stock market, where multiple system failures, for any reason, have a significant financial

impact. For these types of installations, and many more like them, it is important to guard not only against single points of failure, but against **multiple points of failure (MPOF)**, or against single massive failures that cause many components to fail, such as the failure of a data center, of an entire site, or of a small area. A **data center**, in the context of disaster recovery, is a physically proximate collection of nodes and disks, usually all in one room.

Creating clusters that are resistant to multiple points of failure or single massive failures requires a different type of cluster architecture called a **disaster tolerant architecture**. This architecture provides you with the ability to fail over automatically to another part of the cluster or manually to a different cluster after certain disasters. Specifically, the disaster tolerant cluster provides appropriate failover in the case where a disaster causes an entire data center to fail, as in Figure 1-2.

Figure 1-2 Disaster Tolerant Architecture



Understanding Types of Disaster Tolerant Clusters

To protect against multiple points of failure, cluster components must be geographically dispersed: nodes can be put in different rooms, on different floors of a building, or even in separate buildings or separate cities. The distance between the nodes is dependent on the types of disaster from which you need protection, and on the technology used to replicate data. Three types of disaster-tolerant clusters are described in this guide:

- Extended Distance Clusters
- Cluster Extension (CLX) Cluster
- Continental Cluster

These types differ from a simple local cluster in many ways. Extended distance clusters and metropolitan clusters often require right-of-way from local governments or utilities to lay network and data replication cables or connect to DWDMs. This can complicate the design and implementation. They also require a different kind of control mechanism for ensuring that data integrity issues do not arise, such as a quorum server. Typically, extended distance and metropolitan clusters use an arbitrator site containing a computer running a “quorum” application. Continental clusters span great distances and operate by replicating data between two completely separate local clusters.

NOTE Continental clusters are not supported with HP Serviceguard for Linux. They are described here to show the range of solutions that exist.

Extended Distance Clusters

An **extended distance cluster** (also known as **extended campus cluster**) is a normal Serviceguard cluster that has alternate nodes located in different data centers separated by some distance, with a third location supporting the quorum service. Extended distance clusters are connected using a high speed cable that guarantees network access between the nodes as long as all guidelines for disaster tolerant

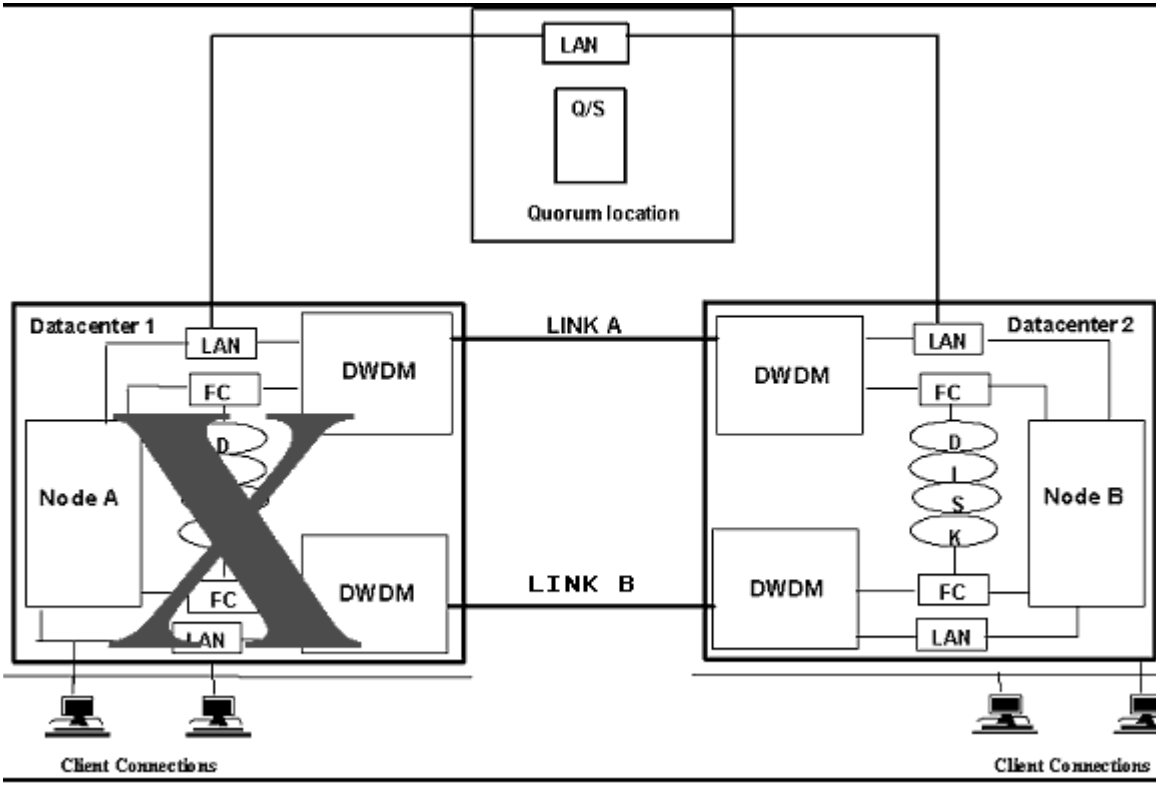
architecture are followed. Extended distance clusters were formerly known as **campus clusters**, but that term is not always appropriate because the supported distances have increased beyond the typical size of a single corporate campus. The maximum distance between nodes in an extended distance cluster is set by the limits of the data replication technology and networking limits. An extended distance cluster is shown in Figure 1-3.

NOTE There are no rules or recommendations on how far the third location must be from the two main data centers. The third location can be as close as the room next door with its own power source or can be as far as in a site across town. The distance among all three locations dictates the level of disaster tolerance an extended distance cluster can provide.

In an extended distance cluster, for data replication, the Multiple Disk (MD) driver is used. Using the MD kernel driver, you can configure RAID 1 (mirroring) in your cluster. In a dual data center setup, to configure RAID 1, one LUN from a storage device in data center 1 is coupled with a LUN from a storage device in data center 2. As a result, the data that is written to this MD device is simultaneously written to both devices. A package that is running on one node in one data center has access data from both storage devices.

The two recommended configurations for the extended distance cluster are both described below.

Figure 1-3 Extended Distance Cluster



In the above configuration the network and FC links between the data centers are combined and sent over common DWDM links. Two DWDM links provide redundancy. When one of them fails, the other may still be active and may keep the two data centers connected. Using the DWDM link, clusters can now be extended to greater distances which were not possible earlier due to limits imposed by the Fibre Channel link for storage and Ethernet for networks. Storage in both data centers is connected to both the nodes via two FC switches in order to provide multiple paths. This configuration supports a distance up to 100 kms between datacenter1 and datacenter2.

Figure 1-4 Two Data Center Setup

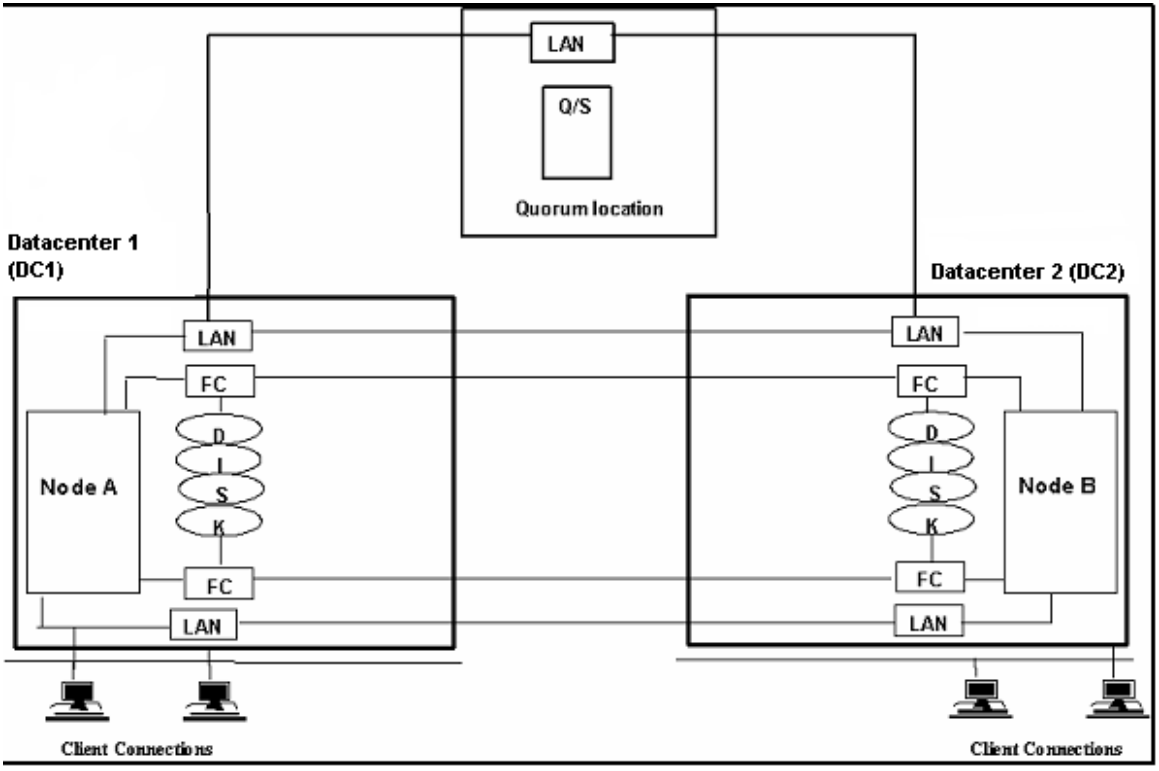


Figure 1-4 shows a configuration that is supported with separate network and FC links between the data centers. In this configuration, the FC links and the Ethernet networks are not carried over DWDM links. But each of these links is duplicated between the two data centers, for redundancy. The disadvantage of having the network and the FC links separate is that if there is a link failure between sites, the ability to exchange heartbeats and the ability to write mirrored data will not be lost at the same time. This configuration is supported to a distance of 10 kms between data centers.

All the nodes in the extended distance cluster must be configured with QLogic driver's multipath feature in order to provide redundancy in connectivity to the storage devices. Mirroring for the storage is configured such that each half of the mirror (disk set) will be physically present at one datacenter each. Further, from each of the nodes there are multiple paths to both of these mirror halves.

Also note that the networking in the configuration shown is the minimum. Added network connections for additional heartbeats are recommended.

Benefits of Extended Distance Cluster

- This configuration implements a single Serviceguard cluster across two data centers, and uses either Multiple Device (MD) driver for data replication.
- You may choose any mix of Fibre Channel-based storage supported by Serviceguard, that also supports the QLogic multipath feature.
- This configuration may be the easiest to understand, as it is similar in many ways to a standard Serviceguard cluster.
- Application failover is minimized. All disks are available to all nodes, so that if a primary disk fails but the node stays up and the replica is available, there is no failover (that is, the application continues to run on the same node while accessing the replica).
- Data copies are peers, so there is no issue with reconfiguring a replica to function as a primary disk after failover.
- Writes are synchronous, so data remains current between the primary disk and its replica, unless the link or disk is down.

Cluster Extension (CLX) Cluster

A Linux CLX cluster is similar to an HP-UX **metropolitan cluster** and is a cluster that has alternate nodes located in different parts of a city or in nearby cities. Putting nodes further apart increases the likelihood that alternate nodes will be available for failover in the event of a disaster. The architectural requirements are the same as for an extended distance cluster, with the additional constraint of a third location for arbitrator node(s) or quorum server. And as with an extended distance cluster, the distance separating the nodes in a metropolitan cluster is limited by the data replication and network technology available.

In addition, there is no hard requirement on how far the third location has to be from the two main data centers. The third location can be as close as the room next door with its own power source or can be as far as in a site across town. The distance between all three locations dictates the level of disaster tolerance a metropolitan cluster can provide.

On Linux, the metropolitan cluster is implemented using CLX.

- CLX for XP
- CLX for EVA

For HP-UX, Metropolitan cluster architecture is implemented through the following HP products:

- Metrocluster with Continuous Access XP
- Metrocluster with Continuous Access EVA
- Metrocluster with EMC SRDF

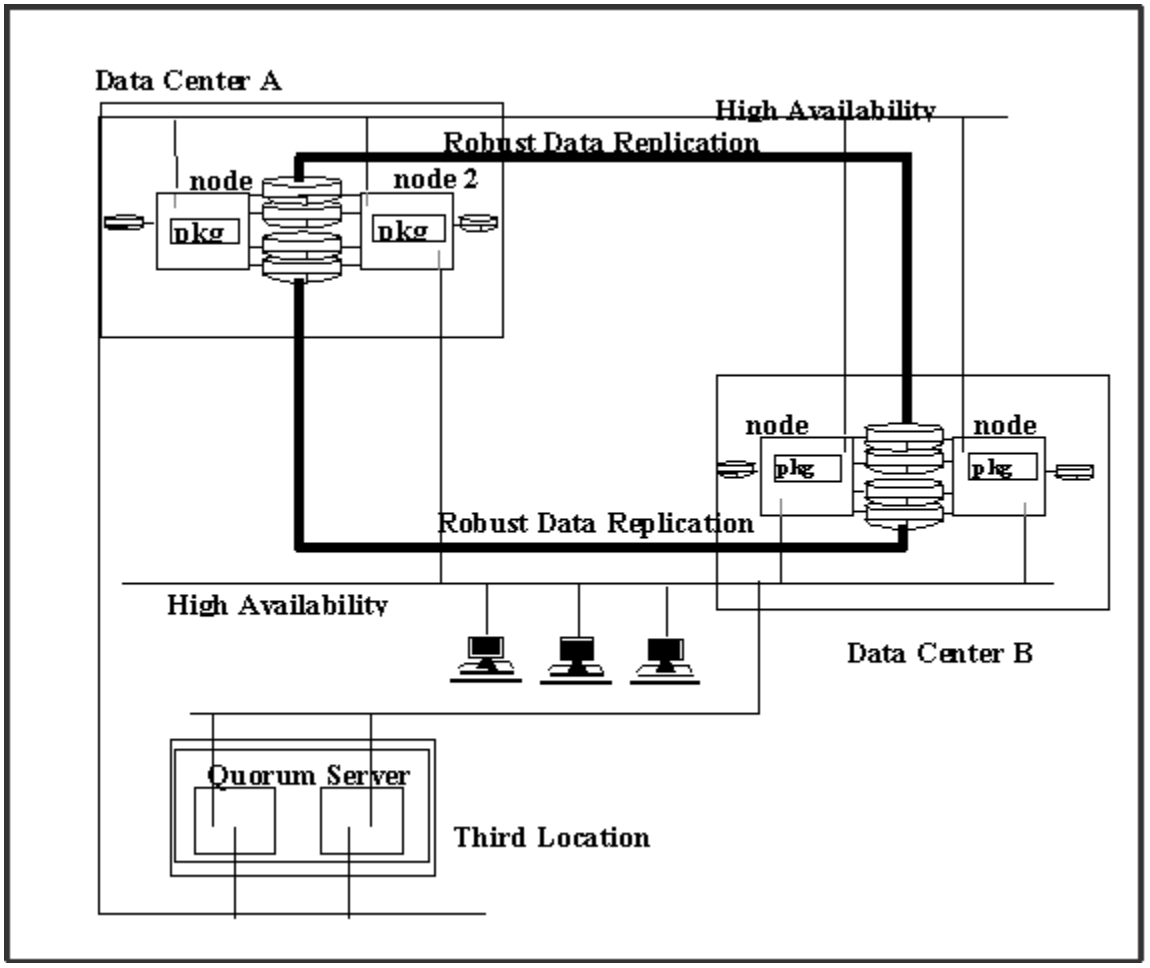
The above HP-UX products are described in detail in Chapters 3, 4, and 5 of the *Designing Disaster Tolerant HA Clusters Using Metrocluster and Continentalclusters* user's guide. The Linux products are described in detail in *Getting Started with MC/ServiceGuard for Linux* guide. While there are some differences between the HP-UX and the Linux versions, the concepts are similar enough that only Cluster Extension (CLX) will be described here.

On-line versions of the above document and other HA documentation are available at **`http://docs.hp.com`** -> High Availability.

On-line versions of the Cluster Extension documentation is available at **`http://h71028.www7.hp.com/enterprise/cache/120851-0-0-225-121.html`**-> HP StorageWorks Cluster Extension EVA or XP.

Figure 1-5 shows a CLX for a Linux Serviceguard cluster architecture.

Figure 1-5 CLX for Linux Serviceguard Cluster



A key difference between extended distance clusters and CLX clusters is the data replication technology used. The extended distance cluster uses Fibre Channel and Linux MD software mirroring for data replication. CLX clusters provide extremely robust hardware-based data replication available with specific disk arrays based on the capabilities of the HP StorageWorks Disk Array XP series, or the HP StorageWorks EVA disk arrays.

Benefits of CLX

- CLX offers a more resilient solution than Extended Distance Cluster, as it provides complete integration between Serviceguard's application package and the data replication subsystem. The storage subsystem is queried to determine the state of the data on the arrays.

  CLX knows that application package data is replicated between two data centers. It takes advantage of this knowledge to evaluate the status of the local and remote copies of the data, including whether the local site holds the primary copy or the secondary copy of data, whether the local data is consistent or not and whether the local data is current or not. Depending on the result of this evaluation, CLX decides if it is safe to start the application package, whether a resynchronization of data is needed before the package can start, or whether manual intervention is required to determine the state of the data before the application package is started.

  CLX allows for customization of the startup behavior for application packages depending on your requirements, such as data currency or application availability. This means that by default, CLX will always prioritize data consistency and data currency over application availability. If, however, you choose to prioritize availability over currency, you can configure CLX to start up even when the state of the data cannot be determined to be fully current (but the data is consistent).
- CLX XP supports synchronous and asynchronous replication modes, allowing you to prioritize performance over data currency between the data centers.
- Because data replication and resynchronization are performed by the storage subsystem, CLX may provide significantly better performance than Extended Distance Cluster during recovery. Unlike Extended Distance Cluster, CLX does not require any additional CPU time for data replication, which minimizes the impact on the host.
- There is little or no lag time writing to the replica, so the data remains current.
- Data can be copied in both directions, so that if the primary site fails and the replica takes over, data can be copied back to the primary site when it comes back up.

- Disk resynchronization is independent of CPU failure (that is, if the hosts at the primary site fail but the disk remains up, the disk knows it does not have to be resynchronized).

Differences Between Extended Distance Cluster and CLX

The *major* differences between an Extended Distance Cluster and a CLX cluster are:

- The methods used to replicate data between the storage devices in the two data centers. The two basic methods available for replicating data between the data centers for Linux clusters are either host-based or storage array-based. Extended Distance Cluster always uses host-based replication (MD mirroring on Linux). Any (mix of) Serviceguard supported Fibre Channel storage can be implemented in an Extended Distance Cluster. CLX always uses array-based replication/mirroring, and requires storage from the same vendor in both data centers (that is, a pair of XPs with Continuous Access, or a pair of EVAs with Continuous Access).
- Data centers in an Extended Distance Cluster can span up to 100km, whereas the distance between data centers in a Metrocluster is defined by the *shortest* of the following distances:
  - Maximum distance that guarantees a network latency of no more than 200ms
  - Maximum distance supported by the data replication link
  - Maximum supported distance for DWDM as stated by the provider
- In an Extended Distance Cluster, there is no built-in mechanism for determining the state of the data being replicated. When an application fails over from one data center to another, the package is allowed to start up if the volume group(s) can be activated. A CLX implementation provides a higher degree of data integrity; that is, the application is only allowed to start up based on the state of the data and the disk arrays.

  It is possible for data to be updated on the disk system local to a server running a package without remote data being updated. This happens if the data link between sites is lost, usually as a precursor to a site going down. If that occurs and the site with the latest data then goes down, that data is lost. The period of time from the link lost to the site going down is called the "recovery point". An

"objective" can be set for the recovery point such that if data is updated for a period less than the objective, automated failover can occur and a package will start. If the time is longer than the objective, then the package will not start. In a Linux environment, this is a user configurable parameter: RPO_TARGET.

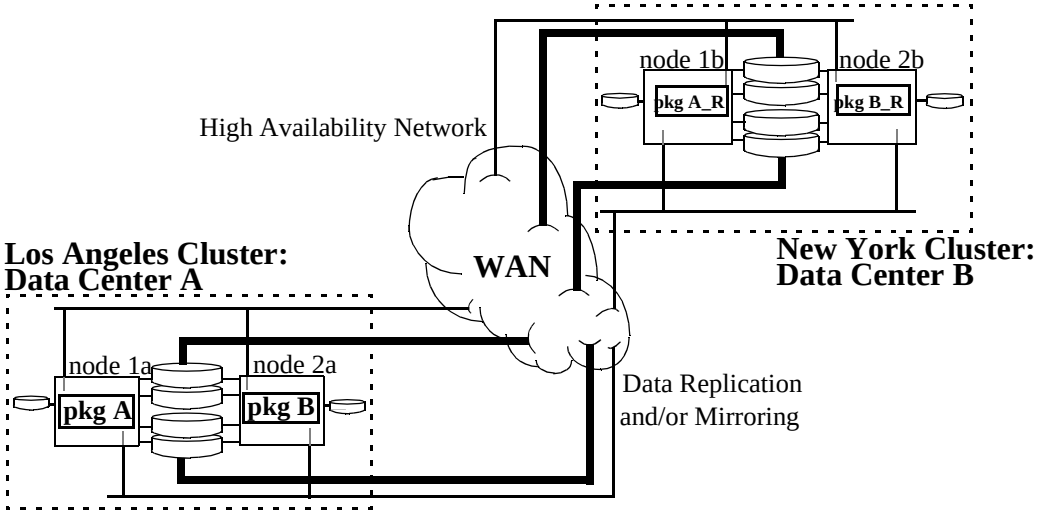
- Extended Distance Cluster disk reads may outperform CLX in normal operations. On the other hand, CLX data resynchronization and recovery performance are better than Extended Distance Cluster.

Continental Cluster

A **continental cluster** provides an alternative disaster tolerant solution in which distinct *clusters* can be separated by large distances, with wide area networking used between them. Continental cluster architecture is implemented using the Continentalclusters product, described fully in Chapter 2 of the *Designing Disaster Tolerant HA Clusters Using Metrocluster and Continentalclusters* user's guide. This product is available only on HP-UX and not on Linux. The design is implemented with two distinct Serviceguard clusters that can be located in different geographic areas with the same or different subnet configuration. In this architecture, each cluster maintains its own quorum, so an arbitrator data center is not used for a continental cluster. A continental cluster can use any WAN connection through a TCP/IP protocol; however, due to data replication needs, high speed connections such as T1 or T3/E3 leased lines or switched lines may be required. See Figure 1-6.

NOTE A continental cluster can also be built using multiple clusters that communicate over shorter distances using a conventional LAN.

Figure 1-6 Continental Cluster



Continentalclusters provides the flexibility to work with any data replication mechanism. It provides pre-integrated solutions that use HP StorageWorks Continuous Access XP, HP StorageWorks Continuous Access EVA, or EMC Symmetrix Remote Data Facility for data replication.

The points to consider when configuring a continental cluster over a WAN are:

- Inter-cluster connections are TCP/IP based.
- The physical connection is one or more leased lines managed by a common carrier. Common carriers cannot guarantee the same reliability that a dedicated physical cable can. The distance can introduce a time lag for data replication, which creates an issue with data currency. This could increase the cost by requiring higher speed WAN connections to improve data replication performance and reduce latency.
- Operational issues, such as working with different personnel trained on different processes, and conducting failover rehearsals, are made more difficult the further apart the nodes are in the cluster.

Benefits of Continentalclusters

- You can virtually build data centers anywhere and still have the data centers provide disaster tolerance for each other. Since Continentalclusters uses two clusters, theoretically there is no limit to the distance between the two clusters. The distance between the clusters is dictated by the required rate of data replication to the remote site, level of data currency, and the quality of networking links between the two data centers.
- In addition, inter-cluster communication can be implemented with either a WAN or LAN topology. LAN support is advantageous when you have data centers in close proximity to each other, but do not want the data centers configured into a single cluster. One example may be when you already have two Serviceguard clusters close to each other and, for business reasons, you cannot merge these two clusters into a single cluster. If you are concerned with one of the centers becoming unavailable, Continentalclusters can be added to provide disaster tolerance. Furthermore, Continentalclusters can be implemented with an existing Serviceguard cluster architecture while keeping both clusters running, and provide flexibility by supporting disaster recovery failover between two clusters that are on the same subnet or on different subnets.
- You can integrate Continentalclusters with any storage component of choice that is supported by Serviceguard. Continentalclusters provides a structure to work with any type of data replication mechanism. A set of guidelines for integrating other data replication schemes with Continentalclusters is included in the *Designing Disaster Tolerant HA Clusters Using Metrocluster and Continentalclusters* user's guide.
- Besides selecting your own storage and data replication solution, you can also take advantage of the following HP pre-integrated solutions:
  - Storage subsystems implemented by CLX are also pre-integrated with Continentalclusters. Continentalclusters uses the same data replication integration module that CLX implements to check for data status of the application package before package start up.
  - If Oracle DBMS is used and logical data replication is the preferred method, depending on the version, either Oracle 8i Standby or Oracle 9i Data Guard with log shipping is used to

replicate the data between two data centers. HP provides a supported integration toolkit for Oracle 8i Standby DB in the Enterprise Cluster Management Toolkit (ECMT).

- RAC is supported by Continentalclusters by integrating it with SGeRAC. In this configuration, multiple nodes in a single cluster can simultaneously access the database (that is, nodes in one data center can access the database). If the site fails, the RAC instances can be recovered at the second site.
- Continentalclusters supports a maximum of 4 clusters with up to 16 nodes per cluster (for a maximum of 64 nodes) supporting up to 3 primary clusters and one recovery cluster.
- Failover for Continentalclusters is semi-automatic. If a data center fails, the administrator is advised, and is required to take action to bring the application up on the surviving cluster.

Continental Cluster With Cascading Failover

A continental cluster with **cascading failover** uses three main data centers distributed between a metropolitan cluster, which serves as a primary cluster, and a standard cluster, which serves as a recovery cluster.

Cascading failover means that applications are configured to fail over from one data center to another in the primary cluster and then to a third (recovery) cluster if the entire primary cluster fails. Data replication also follows the cascading model. Data is replicated from the primary disk array to the secondary disk array in the Metrocluster, then replicated to the third disk array in the Serviceguard recovery cluster.

For more information on Cascading Failover configuration, maintenance, and recovery procedures, see the "*Cascading Failover in a Continental Cluster*" white paper on the high availability documentation web site at **`http://docs.hp.com`** -> High Availability -> Continentalcluster.

Comparison of Disaster Tolerant Solutions

Table 1-1 summarizes and compares the disaster tolerant solutions that are currently available:

Table 1-1 Comparison of Disaster Tolerant Cluster Solutions

Attributes	Extended Distance Cluster	CLX	Continentalclusters (HP-UX only)
Key Benefit	Excellent in "normal" operations, and partial failure. Since all hosts have access to both disks, in a failure where the node is running and the application is up, but the disk becomes unavailable, no failover occurs. The node will access the remote disk to continue processing.	Two significant benefits: • Provides maximum data protection. State of the data is determined before application is started. If necessary, data resynchronization is performed before application is brought up. • Better performance than Extended Distance Cluster for resynchronization, as replication is done by storage subsystem (no impact to host).	Increased data protection by supporting unlimited distance between data centers (protects against such disasters as those caused by earthquakes or violent attacks, where an entire area can be disrupted).

Table 1-1 Comparison of Disaster Tolerant Cluster Solutions (Continued)

Attributes	Extended Distance Cluster	CLX	Continentalclusters (HP-UX only)
Key Limitation	No ability to check the state of the data before starting up the application. If the volume group (vg) can be activated, the application will be started. If mirrors are split or multiple paths to storage are down, as long as the vg can be activated, the application will be started. Data resynchronization does not have a big impact on system performance. However, the performance varies depending on the number of times data resynchronization occurs. In the case of MD, data resynchronization is done one disk at a time, using about 10% of the available CPU time and taking longer to resynchronize multiple LUNs. The amount of CPU time used is a configurable MD parameter.	Specialized storage required. Currently, XP with Continuous Access, and EVA with Continuous Access are supported.	No automatic failover between clusters.

Table 1-1 Comparison of Disaster Tolerant Cluster Solutions (Continued)

Attributes	Extended Distance Cluster	CLX	Continentalclusters (HP-UX only)
Maximum Distance	100 Kilometers	Shortest of the distances between: • Cluster network latency (not to exceed 200 ms). • Data Replication Max Distance. • DWDM provider max distance.	No distance restrictions.
Data Replication mechanism	Host-based, through MD. Replication can affect performance (writes are synchronous). Resynchronization can impact performance. (Complete resynchronization is required in many scenarios that have multiple failures.)	Array-based, through Continuous Access XP or Continuous Access EVA. Replication and resynchronization performed by the storage subsystem, so the host does not experience a performance hit. Incremental resynchronizations are done, based on bitmap, minimizing the need for full re-syncs.	You have a choice of either selecting their own SG-supported storage and data replication mechanism, or implementing one of HP's pre-integrated solutions (including Continuous Access XP, Continuous Access EVA, and EMC SRDF for array-based, or Oracle 8i Standby for host based.) Also, you may choose Oracle 9i Data Guard as a host-based solution. Contributed (that is, unsupported) integration templates for Oracle 9i.

Table 1-1 Comparison of Disaster Tolerant Cluster Solutions (Continued)

Attributes	Extended Distance Cluster	CLX	Continentalclusters (HP-UX only)
Application Failover type	Automatic (no manual intervention required).	Automatic (no manual intervention required).	Semi-automatic (user must “push the button” to initiate recovery).
Access Mode for a package	Active/Standby	Active/Standby	Active/Standby
Client Transparency	Client detects the lost connection. You must reconnect once the application is recovered at second site.	Client detects the lost connection. You must reconnect once the application is recovered at second site.	You must reconnect once the application is recovered at second site.
Maximum Cluster Size Allowed	2 nodes for this release.	2 to 16 nodes	1 to 16 nodes in each cluster supporting up to 3 primary clusters and one recovery cluster. (maximum total of 4 clusters-64 nodes)
Storage	Identical storage is not required (replication is host-based with MD mirroring).	Identical Storage is required.	Identical storage is required if storage-based mirroring is used. Identical storage is not required for other data replication implementations.

Table 1-1 Comparison of Disaster Tolerant Cluster Solutions (Continued)

Attributes	Extended Distance Cluster	CLX	Continentalclusters (HP-UX only)
Data Replication Link	Dark Fiber	Dark Fiber Continuous Access over IP Continuous Access over ATM	WAN LAN Dark Fiber (pre-integrated solution) Continuous Access over IP (pre-integrated solution) Continuous Access over ATM (pre-integrated solution)
Cluster Network	Single or multiple IP subnet	Single or multiple IP subnet	Two configurations: Single IP subnet for both clusters (LAN connection between clusters) Two IP subnets – one per cluster (WAN connection between clusters)

Table 1-1 Comparison of Disaster Tolerant Cluster Solutions (Continued)

Attributes	Extended Distance Cluster	CLX	Continentalclusters (HP-UX only)
DTS Software/ Licenses Required	SGLX + XDC	SGLX + CLX XP or CLX EVA	SG + Continentalclusters + (Metrocluster Continuous Access XP or Metrocluster Continuous Access EVA or Metrocluster EMC SRDF or Enterprise Cluster Master Toolkit) or Customer-selected data replication subsystem CC with RAC: SG + SGeRAC + Continentalclusters

Disaster Tolerant Architecture Guidelines

Disaster tolerant architectures represent a shift away from the massive central data centers and towards more distributed data processing facilities. While each architecture will be different to suit specific availability needs, there are a few basic guidelines for designing a disaster tolerant architecture so that it protects against the loss of an entire data center:

- Protecting nodes through geographic dispersion
- Protecting data through replication
- Using alternative power sources
- Creating highly available networks

These guidelines are in addition to the standard high-availability guidelines of redundant components such as multiple paths to storage, network cards, power supplies, and disks.

Protecting Nodes through Geographic Dispersion

Redundant nodes in a disaster tolerant architecture must be geographically dispersed. If they are in the same data center, it is not a disaster tolerant architecture. Figure 1-2 on page 17 shows a cluster architecture with nodes in two data centers: A and B. If all nodes in data center A fail, applications can fail over to the nodes in data center B and continue to provide clients with service.

Depending on the type of disaster you are protecting against and on the available technology, the nodes can be as close as another room in the same building, or as far away as another city. The minimum recommended dispersion is a single building with redundant nodes in different data centers using different power sources. Specific architectures based on geographic dispersion are discussed in the following chapter.

Protecting Data through Replication

The most significant losses during a disaster are the loss of access to data, and the loss of data itself. You protect against this loss through data replication, that is, creating extra copies of the data. Data replication should:

- Ensure **data consistency** by replicating data in a logical order so that it is immediately usable or recoverable. Inconsistent data is unusable and is not recoverable for processing. Consistent data may or may not be current.
- Ensure **data currency** by replicating data quickly so that a replica of the data can be recovered to include all committed disk writes that were applied to the local disks.
- Ensure **data recoverability** so that there is some action that can be taken to make the data consistent, such as applying logs or rolling a database.
- Minimize **data loss** by configuring data replication to address consistency, currency, and recoverability.

Different data replication methods have different advantages with regards to data consistency and currency. Your choice of which data replication methods to use will depend on what type of disaster tolerant architecture you require.

Off-line Data Replication

Off-line data replication is the method most commonly used today. It involves two or more data centers that store their data on tape and either send it to each other (through an express service, if need dictates) or store it off-line in a vault. If a disaster occurs at one site, the off-line copy of data is used to synchronize data and a remote site functions in place of the failed site.

Because data is replicated using physical off-line backup, data consistency is fairly high, barring human error or an untested corrupt backup. However, data currency is compromised by the time delay in sending the tape backup to a remote site.

Off-line data replication is fine for many applications for which recovery time is not an issue critical to the business. Although data might be replicated weekly or even daily, recovery could take from a day to a week

depending on the volume of data. Some applications, depending on the role they play in the business, may need to have a faster recovery time, within hours or even minutes.

On-line Data Replication

On-line data replication is a method of copying data from one site to another across a link. It is used when very short recovery time, from minutes to hours, is required. To be able to recover use of a system in a short time, the data at the alternate site must be replicated in real time on all disks.

Data can be replicated either synchronously or asynchronously. **Synchronous replication** requires one disk write to be completed and replicated before another disk write can begin. This method improves the chances of keeping data consistent and current during replication. However, it greatly reduces replication capacity and performance, as well as system response time. **Asynchronous replication** does not require the primary site to wait for one disk write to be replicated before beginning another. This can be an issue with data currency, depending on the volume of transactions. An application that has a very large volume of transactions can get hours or days behind in replication using asynchronous replication. If the application fails over to the remote site, it would start up with data that is not current.

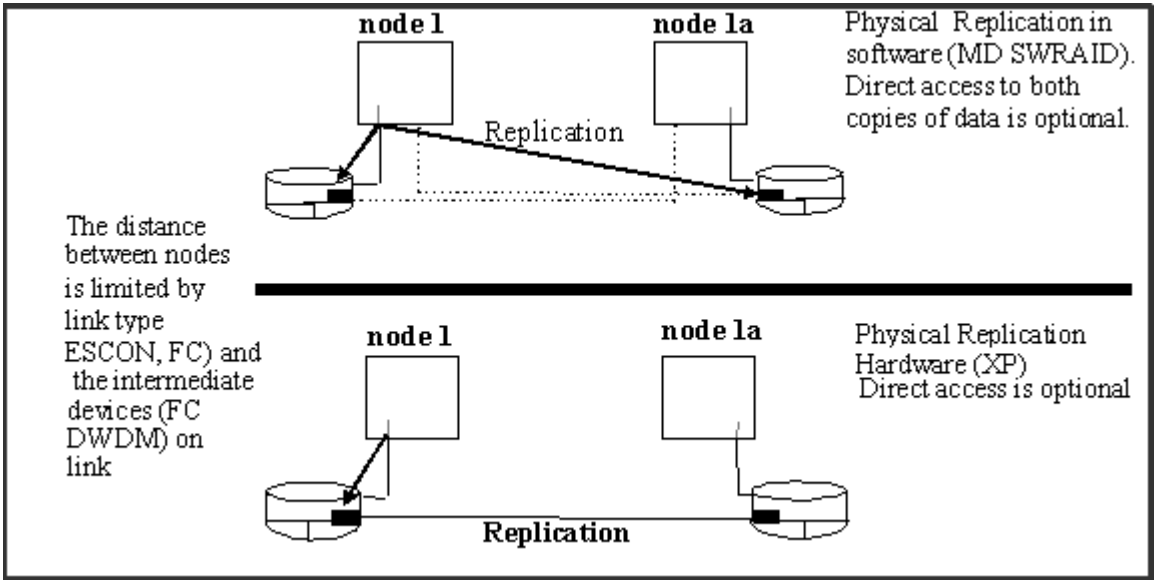
Currently the two ways of replicating data on-line are physical data replication and logical data replication. Either of these can be configured to use synchronous or asynchronous writes.

Physical Data Replication

Each physical write to disk is replicated on another disk at another site. Because the replication is a physical write to disk, it is not application dependent. This allows each node to run different applications under normal circumstances. Then, if a disaster occurs, an alternate node can take ownership of applications and data, provided the replicated data is current and consistent.

As shown in Figure 1-7, physical replication can be done in software or hardware.

Figure 1-7 Physical Data Replication



MD Software RAID is an example of physical replication done in the software; a disk I/O is written to each array connected to the node, requiring the node to make multiple disk I/Os. Continuous Access XP on the HP StorageWorks Disk Array XP series is an example of physical replication in hardware; a single disk I/O is replicated across the Continuous Access link to a second XP disk array.

Advantages of physical replication in hardware are:

- There is little or no lag time writing to the replica. This means that the data remains very current.
- Replication consumes no additional CPU.
- The hardware deals with resynchronization if the link or disk fails. And resynchronization is independent of CPU failure; if the CPU fails and the disk remains up, the disk knows it does not have to be resynchronized.
- Data can be copied in both directions, so that if the primary fails and the replica takes over, data can be copied back to the primary when it comes back up.

Disadvantages of physical replication in hardware are:

- The logical order of data writes is not always maintained in synchronous replication. When a replication link goes down and transactions continue at the primary site, writes to the primary disk are queued in a bit-map. When the link is restored, if there has been more than one write to the primary disk, then there is no way to determine the original order of transactions until the resynchronization has completed successfully. This increases the risk of data inconsistency.

  Also, because the replicated data is a write operation to a physical disk block, database corruption and human errors, such as the accidental removal of a database table, are replicated at the remote site.

NOTE Configuring the disk so that it does not allow a subsequent disk write until the current disk write is copied to the replica (synchronous writes) can limit this risk as long as the link remains up. Synchronous writes impact the capacity and performance of the data replication technology.

- Redundant disk hardware and cabling are required. This, at a minimum, doubles data storage costs, because the technology is in the disk itself and requires specialized hardware.
- For architectures using dedicated cables, the distance between the sites is limited by the cable interconnect technology. Different technologies support different distances and provide different “data through” performance.
- For architectures using common carriers, the costs can vary dramatically, and the connection can be less reliable, depending on the Service Level Agreement.

Advantages of physical replication in software are:

- There is little or no time lag between the initial and replicated disk I/O, so data remains very current.
- The solution is independent of disk technology, so you can use any supported disk technology.
- Data copies are peers, so there is no issue with reconfiguring a replica to function as a primary disk after failover.

- Because there are multiple read devices, that is, the node has access to both copies of data, there may be improvements in read performance.
- Writes are synchronous unless the link or disk is down.

Disadvantages of physical replication in software are:

- As with physical replication in the hardware, the logical order of data writes is not maintained. When the link is restored, if there has been more than one write to the primary disk, there is no way to determine the original order of transactions until the resynchronization has completed successfully.

NOTE Configuring the software so that a write to disk must be replicated on the remote disk before a subsequent write is allowed can limit the risk of data inconsistency while the link is up.

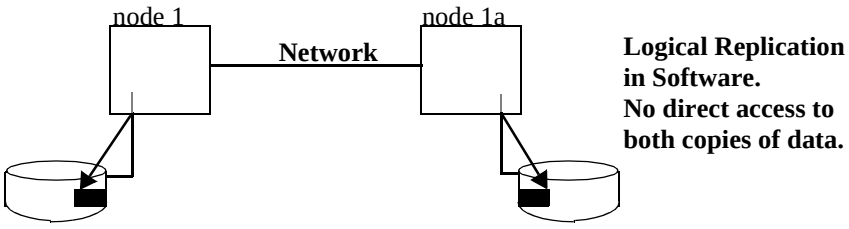
- Additional hardware is required for the cluster.
- Distance between sites is limited by the physical disk link capabilities.
- Performance is affected by many factors: CPU overhead for mirroring, double I/Os, degraded write performance, and CPU time for resynchronization. In addition, CPU failure may cause a resynchronization even if it is not needed, further affecting system performance.

Logical Data Replication

Logical data replication is a method of replicating data by repeating the sequence of transactions at the remote site. Logical replication often must be done at both the file system level, and the database level in order to replicate all of the data associated with an application. Most database vendors have one or more database replication products. An example is the Oracle Standby Database.

Logical replication can be configured to use synchronous or asynchronous writes. Transaction processing monitors (TPMs) can also perform logical replication.

Figure 1-8 Logical Data Replication



Advantages of using logical replication are:

- The distance between nodes is limited only by the networking technology.
- There is no additional hardware needed to do logical replication, unless you choose to boost CPU power and network bandwidth.
- Logical replication can be implemented to reduce risk of duplicating human error. For example, if a database administrator erroneously removes a table from the database, a physical replication method will duplicate that error at the remote site as a raw write to disk. A logical replication method can be implemented to delay applying the data at a remote site, so such errors would not be replicated at the remote site. This also means that administrative tasks, such as adding or removing database tables, has to be repeated at each site.
- With database replication you can roll transactions forward or backward to achieve the level of currency desired on the replica, although this functionality is not available with file system replication.

Disadvantages of logical replication are:

- It uses significant CPU overhead because transactions are often replicated more than once and logged to ensure data consistency, and all but the most simple database transactions take significant CPU. It also uses network bandwidth, whereas most physical replication methods use a separate data replication link. As a result, there may be a significant lag in replicating transactions at the remote site, which affects data currency.

- If the primary database fails and is corrupt, which results in the replica taking over, then the process for restoring the primary database so that it can be used as the replica is complex. This often involves recreating the database and doing a database dump from the replica.
- Applications often have to be modified to work in an environment that uses a logical replication database. Logic errors in applications or in the RDBMS code itself that cause database corruption will be replicated to remote sites. This is also an issue with physical replication.
- Most logical replication methods do not support personality swapping, which is the ability after a failure to allow the secondary site to become the primary and the original primary to become the new secondary site. This capability can provide increased up time.

Ideal Data Replication

The ideal disaster tolerant architecture, if budgets allow, is the following combination:

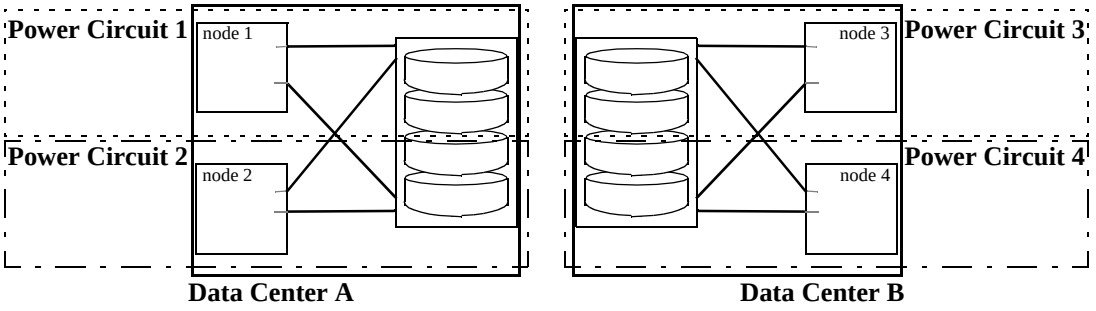
- For performance and data currency—physical data replication.
- For data consistency—either a second physical data replication as a point-in-time snapshot or logical data replication, which would only be used in the cases where the primary physical replica was corrupt.

Using Alternative Power Sources

In a high-availability cluster, redundancy is applied to cluster components, such as multiple paths to storage, redundant network cards, power supplies, and disks. In disaster tolerant architectures another level of protection is required for these redundancies.

Each data center that houses part of a disaster tolerant cluster should be supplied with power from a different circuit. In addition to a standard UPS (uninterrupted power supply), each node in a disaster tolerant cluster should be on a separate power circuit; see Figure 1-9.

Figure 1-9 Alternative Power Sources



Housing remote nodes in another building often implies they are powered by a different circuit, so it is especially important to make sure all nodes are powered from a different source if the disaster tolerant cluster is located in two data centers in the same building. Some disaster tolerant designs go as far as making sure that their redundant power source is supplied by a different power substation on the grid. This adds protection against large-scale power failures, such as brown-outs, sabotage, or electrical storms.

Creating Highly Available Networking

Standard high-availability guidelines require redundant networks. Redundant networks may be highly available, but they are not disaster tolerant if a single accident can interrupt both network connections. For example, if you use the same trench to lay cables for both networks, you do not have a disaster tolerant architecture because a single accident, such as a backhoe digging in the wrong place, can sever both cables at once, making automated failover during a disaster impossible.

In a disaster tolerant architecture, the reliability of the network is paramount. To reduce the likelihood of a single accident causing both networks to fail, redundant network cables should be installed so that they use physically different routes for each network. How you route cables will depend on the networking technology you use. Specific guidelines for some network technologies are listed here.

Disaster Tolerant Local Area Networking

Ethernet networks can also be used to connect nodes in a disaster tolerant architecture within the following guidelines:

- Each node is connected to redundant switches and bridges using two Ethernet host adapters. Bridges, repeaters, or other components that convert from copper to fibre cable may be used to span longer distances.

Disaster Tolerant Wide Area Networking

Disaster tolerant networking for continental clusters is directly tied to the data replication method. In addition to the redundant lines connecting the remote nodes, you also need to consider what bandwidth you need to support the data replication method you have chosen. A continental cluster that handles a high number of transactions per minute will not only require a highly available network, but also one with a large amount of bandwidth.

This is a brief discussion of things to consider when choosing the network configuration for your continental cluster. Details on WAN choices and configurations can be found in Continental Cluster documentation available from: **`http://docs.hp.com`** -> High Availability.

- Bandwidth affects the rate of data replication, and therefore the currency of the data should there be the need to switch control to another site. The greater the number of transactions you process, the more bandwidth you will need. The following connection types offer differing amounts of bandwidth:
  - T1 and T3: low end
  - ISDN and DSL: medium bandwidth
  - ATM: high end
- Reliability affects whether or not data replication happens, and therefore the consistency of the data should you need to fail over to the recovery cluster. Redundant leased lines should be used, and should be from two different common carriers, if possible.
- Cost influences both bandwidth and reliability. Higher bandwidth and dual leased lines cost more. It is best to address data consistency issues first by installing redundant lines, then weigh the price of data currency and select the line speed accordingly.

Disaster Tolerant Cluster Limitations

Disaster tolerant clusters have limitations, some of which can be mitigated by good planning. Some examples of MPOF that may not be covered by disaster tolerant configurations:

- Failure of all networks among all data centers — This can be mitigated by using a different route for all network cables.
- Loss of power in more than one data center — This can be mitigated by making sure data centers are on different power circuits, and redundant power supplies are on different circuits. If power outages are frequent in your area, and down time is expensive, you may want to invest in a backup generator.
- Loss of all copies of the on-line data — This can be mitigated by replicating data off-line (frequent backups). It can also be mitigated by taking snapshots of consistent data and storing it on-line; Business Copy XP and EMC Symmetrix BCV (Business Consistency Volumes) provide this functionality and the additional benefit of quick recovery should anything happen to both copies of on-line data.
- A **rolling disaster** is a disaster that occurs before the cluster is able to recover from failure that is not normally considered a “disaster”. An example is a data replication link that fails, then, as it is being restored and data is being resynchronized, a disaster causes an entire data center to fail. The effects of rolling disasters can be mitigated by ensuring that a copy of the data is stored either off-line or on a separate disk that can quickly be mounted. The trade-off is a lack of currency for the data in the off-line copy.

Managing a Disaster Tolerant Environment

In addition to the changes in hardware and software to create a disaster tolerant architecture, there are also changes in the way you manage the environment. Configuration of a disaster tolerant architecture needs to be carefully planned, implemented and maintained. There are additional resources needed, and additional decisions to make concerning the maintenance of a disaster tolerant architecture:

- *Manage it in-house, or hire a service?*

  Hiring a service can remove the burden of maintaining the capital equipment needed to recover from a disaster. Most disaster recovery services provide their own off-site equipment, which reduces maintenance costs. Often the disaster recovery site and equipment are shared by many companies, further reducing cost.

  Managing disaster recovery in-house gives complete control over the type of redundant equipment used and the methods used to recover from disaster, giving you complete control over all means of recovery.

- *Implement automated or manual recovery?*

  Manual recovery costs less to implement and gives more flexibility in making decisions while recovering from a disaster. Evaluating the data and making decisions can add to recovery time, but it is justified in some situations, for example if applications compete for resources following a disaster and one of them has to be halted.

  Automated recovery reduces the amount of time and in most cases eliminates human intervention needed to recover from a disaster. You may want to automate recovery for any number of reasons:

  — Automated recovery is usually faster.

  — Staff may not be available for manual recovery, as is the case with “lights-out” data centers.

  — Reduction in human intervention is also a reduction in human error. Disasters don’t happen often, so lack of practice and the stressfulness of the situation may increase the potential for human error.

  — Automated recovery procedures and processes can be transparent to the clients.

Even if recovery is automated, you may choose to, or need to recover from some types of disasters with manual recovery. A **rolling disaster**, which is a disaster that happens before the cluster has recovered from a previous disaster, is an example of when you may want to manually switch over. If the data link failed, as it was coming up and resynchronizing data, and the data center failed, you would want human intervention to make judgment calls on which site had the most current and consistent data before failing over.

- *Who manages the nodes in the cluster and how are they trained?*

  Putting a disaster tolerant architecture in place without planning for the people aspects is a waste of money. Training and documentation are more complex because the cluster is in multiple data centers.

  Each data center often has its own operations staff with their own processes and ways of working. These operations people will now be required to communicate with each other and coordinate maintenance and failover rehearsals, as well as working together to recover from an actual disaster. If the remote nodes are placed in a "lights-out" data center, the operations staff may want to put additional processes or monitoring software in place to maintain the nodes in the remote location.

  Rehearsals of failover scenarios are important to keep prepared. A written plan should outline rehearsal of what to do in cases of disaster with a minimum recommended rehearsal schedule of once every 6 months, ideally once every 3 months.

- *How is the cluster maintained?*

  Planned downtime and maintenance, such as backups or upgrades, must be more carefully thought out because they may leave the cluster vulnerable to another failure. For example, nodes need to be brought down for maintenance in pairs: one node at each site, so that quorum calculations do not prevent automated recovery if a disaster occurs during planned maintenance.

  Rapid detection of failures and rapid repair of hardware is essential so that the cluster is not vulnerable to additional failures.

  Testing is more complex and requires personnel in each of the data centers. Site failure testing should be added to the current cluster testing plans.

Additional Disaster Tolerant Solutions Information

On-line versions of HA documentation are available at **`http://docs.hp.com`** -> High Availability -> Serviceguard for Linux.

For information on CLX for EVA and XP, see the following document available at **`http://h71028.www7.hp.com/enterprise/cache/120851-0-0-225-121.html`** -> HP StorageWorks Cluster Extension for EVA or XP.

- *HP StorageWorks Cluster Extension EVA user guide*
- *HP StorageWorks Cluster Extension XP for HP Serviceguard for Linux*

2 Building an Extended Distance Cluster Using Serviceguard and Software RAID

Simple Serviceguard clusters are usually configured in a single data center, often in a single room, to provide protection against failures in CPUs, interface cards, and software. Extended Serviceguard clusters are specialized cluster configurations, which allow a single cluster to extend across two separate data centers for increased disaster tolerance. Depending on the type of links employed, distances of up to 100 kms between data centers can be achieved.

This chapter discusses several types of extended distance cluster that use basic Serviceguard technology with software mirroring (using MD Software RAID) and Fibre Channel. Both two data center and three data center architectures are illustrated. This chapter discusses the following:

Types of Data Link for Storage and Networking

Fibre Channel technology lets you increase the distance between the components in an Serviceguard cluster, thus making it possible to design a disaster tolerant architecture. The following table shows some of the distances possible with a few of the available technologies, including some of the Fiber Optic alternatives.

Table 2-1 Link Technologies and Distances

Type of Link	Maximum Distance Supported
Gigabit Ethernet Twisted Pair	50 meters
Short Wave Fiber	500 meters
Long Wave Fiber	10 kilometers
Dense Wave Division Multiplexing (DWDM)	100 kilometers

The development of DWDM technology allows designers to use dark fiber (high speed communication lines provided by common carriers) to extend the distances that were formerly subject to limits imposed by Fibre Channel for storage and Ethernet for network links.

NOTE Increased distance often means increased cost and reduced speed of connection. Not all combinations of links are supported in all cluster types. For a current list of supported configurations and supported distances, see the *HP Configuration Guide*, available through your HP representative.

Two Data Center and Quorum Service Location Architectures

A two data center and Quorum Service location, which is at a third location, have the following configuration requirements:

NOTE There is no hard requirement on how far the Quorum Service location has to be from the two main data centers. It can be as close as the room next door with its own power source or can be as far as in another site across town. The distance between all three locations dictates that level of disaster tolerance a cluster can provide.

- In these solutions, there must be an equal number of nodes in each primary data center, and the third location (known as the arbitrator data center) contains the Quorum Server. LockLUN is not supported in a Disaster Tolerant configuration. In this release, only one node in each data center is supported.
- The Quorum Server is used as a tie-breaker to maintain cluster quorum when all communication between the two primary data centers is lost. The arbitrator data center must be located separately from the primary data centers. For more information about quorum server, see the *Managing Serviceguard* user's guide and the *Serviceguard Quorum Server Release Notes*.
- A minimum of two heartbeat paths must be configured for all cluster nodes. The preferred solution is two separate heartbeat subnets configured in the cluster, each going over a separately routed network path to the other data center. Alternatively, there can be a single dedicated heartbeat subnet with a bonded pair configured for it. Each would go over a separately routed physical network path to the other data centers.
- There can be separate networking and Fibre Channel links between the data centers, or both networking and Fibre Channel can go over DWDM links between the data centers.

- Fibre Channel Direct Fabric Attach (DFA) is recommended over Fibre Channel Arbitrated loop configurations, due to the superior performance of DFA, especially as the distance increases. Therefore Fibre Channel switches are recommended over Fibre Channel hubs.
- Any combination of the following Fibre Channel capable disk arrays may be used: HP StorageWorks 1000 and 1500 series Modular Storage Arrays, HP StorageWorks Enterprise Virtual Arrays, or HP StorageWorks Disk Array XP.
- For disaster tolerance, application data must be mirrored between the primary data centers. You must ensure that the mirror copies reside in different data centers, as the software cannot determine the locations.

NOTE When a failure results in the mirror copies losing synchronization, MD will perform a full resynchronization when both halves of the mirror are available.

- No routing is allowed for the networks between data centers. Routing is allowed to the third data center if a Quorum Server is used in that data center.

The following is a list of recommended arbitration methods for Extended Distance Cluster solutions in order of preference:

- Quorum Server running in a Serviceguard cluster
- Quorum Server

For more information on Quorum Server, see the *Serviceguard Quorum Server Release Notes for Linux*.

Figure 2-1 is an example of a two data center and third location configuration using DWDM, with a quorum server node on the third site.

Figure 2-1 Two Data Centers and Third Location with DWDM and Quorum Server

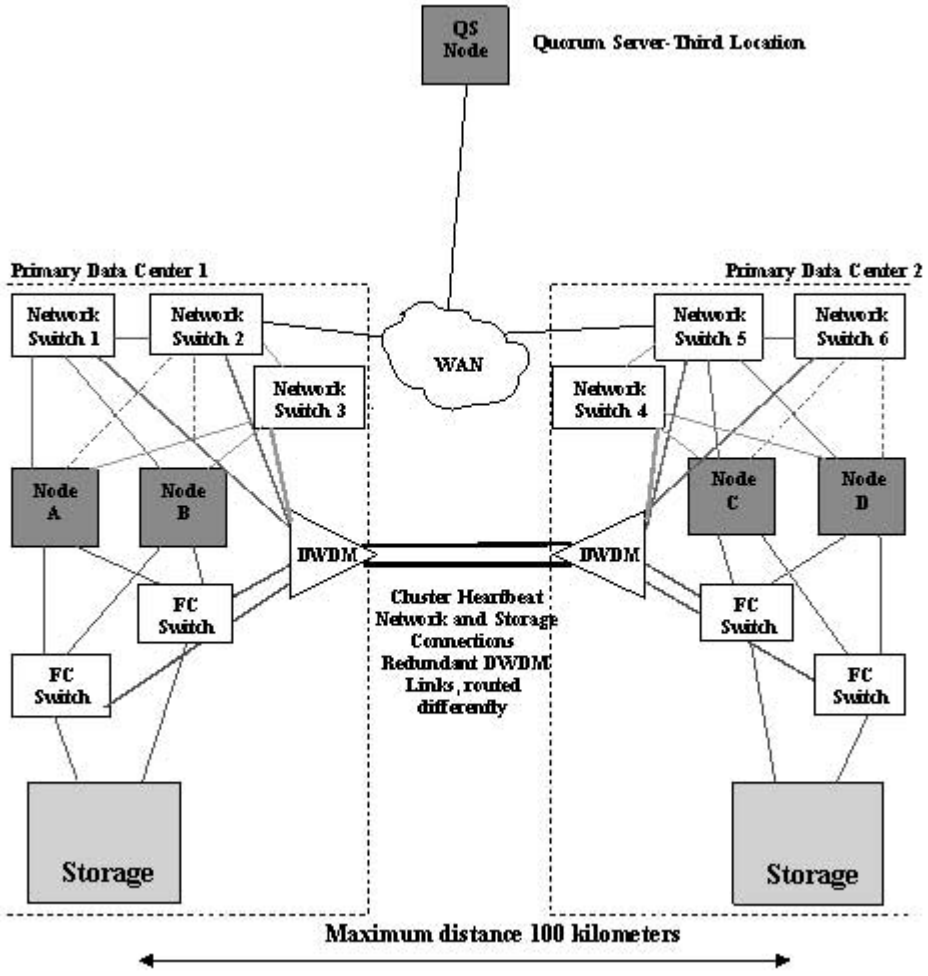


Figure 2-1 is an example of a two data center and third location configuration using DWDM, with a quorum server node on the third site. The DWDM boxes connected between the two Primary Data Centers are configured with redundant dark fiber links and the standby fibre feature has been enabled.

There are no requirements for the distance between the Quorum Server Data center and the Primary Data Centers, however it is necessary to ensure that the Quorum Server can be contacted within a reasonable amount of time (should be within the `NODE_TIMEOUT` period). LockLUN arbitration is not allowed in this configuration.

Rules for Separate Network and Data Links

- There must be less than 200 milliseconds of latency in the network between the data centers.
- No routing is allowed for the networks between the data centers.
- Routing is allowed to the third data center if a Quorum Server is used in that data center.
- The maximum distance between the data centers for this type of configuration is currently limited by the maximum distance supported for the networking type or Fibre Channel link type being used, whichever is shorter.
- There can be a maximum of 500 meters between the Fibre Channel switches in the two data centers if Short-wave ports are used. This distance can be increased to 10 kilometers by using a Long-wave Fibre Channel port on the switches. If DWDM links are used, the maximum distance between the data centers is 100 kilometers. For more information on link technologies, see Table 2-1 on page 52.
- There must be at least two alternately routed networking links between each primary data center to prevent the “backhoe problem”. The “backhoe problem” can occur when all cables are routed through a single trench and a tractor on a construction job severs all cables and disables all communications between the data centers. It is allowable to have only a single network link routed from each primary data center to the third location, however in order to survive the loss of the network link between a primary data center and the arbitrator data center, the network routing should be configured so that a primary data center can also reach the arbitrator via a route which passes through the other primary data center.
- There must be at least two alternately routed Fibre Channel Data Replication links between each data center.
- See the *HP Configuration Guide* (available through your HP representative) for a list of supported Fibre Channel hardware.

Guidelines on DWDM Links for Network and Data

- There must be less than 200 milliseconds of latency in the network between the data centers.
- No routing is allowed for the networks between the data centers.
- Routing is allowed to the third data center if a Quorum Server is used in that data center.
- The maximum distance supported between the data centers for DWDM configurations is 100 kilometers.
- Both the networking and Fibre Channel Data Replication can go through the same DWDM box - separate DWDM boxes are not required.
- Since DWDM converters are typically designed to be fault tolerant, it is acceptable to use only one DWDM box (in each data center) for the links between each data center. However, for the highest availability, it is recommended to use two separate DWDM boxes (in each data center) for the links between each data center. If using a single DWDM box for the links between each data center the redundant standby fibre link feature of the DWDM box must be configured. If the DWDM box supports multiple active DWDM links, that feature can be used instead of the redundant standby feature.
- At least two dark fiber optic links are required between each Primary data center, each fibre link routed differently to prevent the “backhoe problem.” It is allowable to have only a single fibre link routed from each Primary data center to the third location, however in order to survive the loss of a link between a Primary data center and the third data center, the network routing should be configured so that a Primary data center can also reach the Arbitrator via a route passing through the other Primary data center.
- The network switches in the configuration must support DLPI (link level) packets. The network switch can be 100BaseT (TX or FX), 1000BaseT (TX or FX) or FDDI. The connection between the network switch and the DWDM box must be fiber optic.

- Fibre Channel switches must be used in a DWDM configuration; Fibre Channel hubs are not supported. Direct Fabric Attach mode must be used for the ports connected to the DWDM link.

See the *HP Configuration Guide*, available through your HP representative, for more information on supported devices.

3 Configuring your Environment for Software RAID

The previous chapters discussed conceptual information on disaster tolerant architectures and procedural information on creating an extended distance cluster. This chapter discusses the procedures you need to follow to configure Software RAID in your extended distance cluster. Following are the topics discussed in this chapter:

Understanding Software RAID

Redundant Array of Independent Disks (RAID) is a mechanism that provides storage fault tolerance and, occasionally, better performance. Software RAID is designed on the concept of RAID 1. RAID 1 uses mirroring where data is written to two disks at the same time.

The Serviceguard XDC product uses the Multiple Device (MD) driver and its associated tool mdadm to implement Software RAID. With Software RAID, two disks (or disk sets) are configured so that the same data is written on both disks as one "write transaction". So if data from one disk set is lost, or if one disk set is rendered unavailable, the data is always available from the second disk set. As a result, high availability of data is guaranteed. In an extended distance cluster, the two disk sets are in two physically separated locations, so if one location becomes unavailable, the other location still has the data.

For more information on Linux Software RAID, see *The Software-RAID HOWTO* manual available at:
`http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html`

To enable Software RAID in your extended distance cluster, you need to complete the following:

1. Install the extended distance cluster software.
2. Copy the files into package directories.
3. Configure packages that will use Software RAID.

The subsequent sections include information on installing Extended Distance Cluster software, and configuring your environment for Software RAID.

Installing the Extended Distance Cluster Software

This section discusses the supported operating systems, prerequisites and the procedures for installing the Extended Distance Cluster software.

Supported Operating Systems

The Extended Distance Cluster software supports the following operating systems:

- Red Hat 4 U3 or later
- Novell SUSE Linux Enterprise Server 9 SP3 or later
- Novell SUSE Linux Enterprise Server 10 or later

Prerequisites

Following are the prerequisites for installing Extended Distance Cluster software (XDC):

- HP Serviceguard for Linux A 11.16.07 or higher
- Network Time Protocol (NTP) - all nodes in the cluster to point to the same NTP server.
- QLogic Driver - The version number of this driver depends on the version of the QLogic cards in your environment. Download the appropriate version of the driver from the following location:

  http://www.hp.com -> Software and Driver Downloads

  Select the Download drivers and software (and firmware) option. Enter the HBA name and click >>. If more than one result is displayed, download the appropriate driver for your operating system.

Installing XDC

You can install XDC from the product CD. You must install the XDC software on all nodes in a cluster to enable Software RAID.

Complete the following procedure to install XDC:

1. Insert the product CD into the drive and mount the CD.
2. Open the command line interface.
3. If you are installing XDC on Red Hat 4, run the following command:

   # rpm -Uvh xdc-A.01.00-0.rhel4.noarch.rpm
4. If you are installing XDC on Novell SUSE Linux Enterprise Server 9, run the following command:

   # rpm -Uvh xdc-A.01.00-0.sles9.noarch.rpm
5. If you are installing XDC on Novell SUSE Linux Enterprise Server 10, run the following command:

   # rpm -Uvh xdc-A.01.00-0.sles10.noarch.rpm

   This command initializes the XDC software installation. After you install XDC, you need to copy the `raid.conf.template` into each package directory that you need to enable Software RAID.
6. Run the following command to copy the `raid.conf.template` file as `raid.conf` file into each package directory:

   **# cp $SGROOT/xdc/raid.conf.template **
   $SGCONF/<pkgdir>/raid.conf

   The file is copied into the package directory.

NOTE Installing the Extended Distance Cluster software does not enable Software RAID for every package in your environment. You need to manually enable Software RAID for a package by copying the files into the package directories. Also, if you need to enable Software RAID for more than one package in your environment, you need to copy the files and templates into each of those package directories. You must edit these template files later.

Verifying the XDC Installation

After you install XDC, run the following command to ensure that the software is installed:

#rpm -qa| grep xdc

In the output, the product name, `xdc -A.01.00-0` will be listed. The presence of this file verifies that the installation is successful.

Configuring the Environment

After setting up the hardware as described in the Extended Distance Cluster Architecture section and installing the Extended Distance Cluster software, complete the following steps to enable Software RAID for each package. Subsequent sections describe each of these processes in detail.

1. Configure multipath for storage

   In the Extended Distance Cluster setup described in figures 1 and 2, a node has multiple paths to storage. With this setup each LUN exposed from a storage array shows up as two devices on every node. There are two device entries in the /dev directory for the same LUN where each device entry will pertain to a single path to that LUN. When a QLogic driver is installed and configured for multipath, all device names leading to the same physical device will be merged and only one device entry will appear in their place. This happens for devices from both the storage systems. Creating these multiple links to the storage device ensures that each node is not dependent only on one link to write data to that storage device. For more information on configuring multipath, see “Configuring Multiple Paths to Storage” on page 69.

2. Configure persistent device names for storage devices

   Once the multipath has been configured, you need to create persistent device names using udev. In cases of disk or link failure and subsequent reboot, it is possible that device names are renamed or reoriented. Since the MD mirror device starts with the names of the component devices, a change in the device name prevents the MD mirror from starting. To avoid this problem, HP requires that you make the device names persistent. For more information on configuring persistent device names, see “Using Persistent Device Names” on page 71.

3. Create the MD mirror device

   To enable Software RAID in your environment, you need to first create the mirror setup. This implies that you specify two disks to create a Multiple Device (MD). When configuring disks in RAID 1 level, use a disk or LUN from each datacenter as one mirror half. Be sure to create disk sets of the same size as they need to store data

that are of identical sizes. Differences in disk set size results in a mirror being created of a size equal to the smaller of the two disks. Be sure to create the mirror using the persistent device names of the component devices. For more information on creating and managing a mirrored device, see “Creating a Multiple Disk Device” on page 72.

4. Create volume groups and logical volumes on the MD mirror device

   Once the MD mirror device is created, you need to create volume groups and logical volumes on it. You must use the Volume Group Exclusive activation feature. This protects against a volume group which is already active on one node to be activated again (accidentally or on purpose) on any other node in the cluster. For more information on creating volume groups and configuring exclusive activation, see “Creating Volume Groups and Configuring VG Exclusive Activation on the MD Mirror” on page 74.

5. Configure the package control script and the Extended Distance cluster configuration script

   In order to let Serviceguard know of the existence of the mirror created in the previous step and hence make use of it, it must be configured as part of a package. This MD device must be specified in the Extended Distance Cluster configuration file `raid.conf`. Copy the `raid.conf.template` in the software bundle as `raid.conf` to the package directory and edit it to specify the RAID configuration parameters for this package. Using the details mentioned in this file Serviceguard will start, stop and monitor this MD mirror for the package. For details on how to configure the package control script and raid.conf see “Configuring the Package Control Script and RAID Configuration File” on page 76.

IMPORTANT

Every time you edit the `raid.conf` file, you must copy this edited file to all nodes in the cluster.

6. Start the package

   Starting a package configured for Software RAID is the same as starting any other package in Serviceguard for Linux.

You also need to keep in mind a few guidelines before you enable Software RAID for a particular package. Following are some of these guidelines you need to follow:

- Ensure that the Quorum Server link is close to the Ethernet links in your setup. In cases of failures of all Ethernet and Fibre channel links, the nodes can easily access the Quorum Server for arbitration.
- The Quorum Server is configured in a third location only for arbitration. In scenarios where the link between two nodes is lost, each node considers the other node to be dead. As a result, both nodes will try to access the Quorum Server. The Quorum Server, as an arbitrator, acknowledges the node that reaches it first and allows the package to start on that node.
- You also need to configure Network Time Protocol (NTP) in your environment. This protocol resolves the time differences that could occur in a network. For example, nodes in different time zones.

Configuring Multiple Paths to Storage

HP requires that you configure multiple paths to the storage device using the QLogic HBA driver as it has inbuilt multipath capabilities. Use the install script with the "-f" option to enable multipath failover mode.

For more information on installing the QLogic HBA driver, see the *HP StorageWorks Using the QLogic HBA driver for single-path or multipath failover mode on Linux systems application notes*. This document is available at the following location:

http://h20000.www2.hp.com/bc/docs/support/SupportManual/c00169487/c00169487.pdf

NOTE You need to register with Hewlett-Packard to access this site.

Setting the Value of the Link Down Timeout Parameter

After you install the QLogic HBA driver, you must set the `Link Down Timeout` parameter of the QLogic cards to a duration equal to the cluster reformation time. When using the default values of heartbeat interval and node timeout intervals of Serviceguard for Linux with a Quorum server, this parameter must be set to 40 seconds. Setting this parameter to 40 seconds, which is the recommended value, prevents further writes to the active half of the mirror disk set when the other fails. If this failure were to also bring down the node a few moments later, then the chance of losing these writes are eliminated.

This parameter prevents any data being written to a disk when a failure occurs. The value of this parameter must be set such that the disks are inaccessible for a time period which is greater than the cluster reformation time. This parameter is important in scenarios where an entire site is in the process of going down. By blocking further writes to the MD device, the two disks of the MD device remain current and synchronized. As a result, when the package fails over, it starts with a disk that has current data. You must set a value for this parameter for all QLogic cards.

The QLogic cards are configured to hold up any disk access and essentially hang for a time period which is greater than the cluster reformation time when access to a disk is lost. This is achieved by altering the Link Down Timeout value for each port of the card. Setting a value for the Link Down Timeout parameter for a QLogic card ensures that the MD device hangs when access to a mirror is lost. For configurations with multipath, the MD device hangs when one path to a storage system is lost. However, the MD device resumes activity when the specified hang period expires. This ensures that no data is lost.

This parameter is required to address a scenario where an entire datacenter fails but all its components do not fail at the same time but undergo a rolling failure. In this case, if the access to one disk is lost, the MD layer hangs and data is no longer written to it. Within the hang period, the node goes down and a cluster reformation takes place. When the package fails over to another node, it starts with a disk that has current data.

The value to be set for `Link Down Timeout` parameter depends on the heartbeat interval and the node timeout values configured for a particular cluster. Use the SANSurfer CLI tool to set the value for this parameter. For more information on how to set this parameter, see **`http://download.qlogic.com/manual/32338/SN0054614-00B.pdf`**

Table 3-1 lists the heartbeat intervals and the node timeout values for a particular cluster.

Table 3-1 Cluster Reformation Time and Timeout Values

Heartbeat Intervals	Cluster Reformation Time	Link Down Timeout Value
1 second	38 seconds	40 seconds
2 seconds	56 seconds	58 seconds
5 seconds	140 seconds	142 seconds
10 seconds	250 seconds	255 seconds

NOTE The values in this table are approximate values. The actual time varies from system to system, depending on the system load.

Using Persistent Device Names

When there is a disk related failure and subsequent reboot, there is a possibility that the devices are renamed. Linux names disks in the order they are found. The device that was `/dev/sdf` may be renamed to `/dev/sde` if any "lower" device is failed or removed. As a result, you cannot activate the MD device with the original name.

HP requires that the device names be persistent to avoid reorientation after a failure and reboot. For more information on creating persistent device names, see the *Using udev to Simplify HP Serviceguard for Linux Configuration* white paper that is available at the following location:

`http://docs.hp.com`

When creating persistent device names, ensure that the same udev rules file exists in all the nodes. This is necessary for the symlinks to appear and point to the correct device. Use these persistent device names wherever there is a need to specify the devices for extended cluster configuration or during recovery process after a failure. A persistent device created based on the instructions in the document mentioned earlier will have a device name that starts with `/dev/hpdev/`.

NOTE The name of the MD device must be unique across all packages in the cluster. Also, the names of each of their component udev devices must also be unique across all nodes in the cluster.

Creating a Multiple Disk Device

As mentioned earlier, the first step for enabling Software RAID in your environment is to create the Multiple Disk (MD) device using two underlying component disks. This MD device is a virtual device which ensures that any data written to it is written to both component disks. As a result, the data is identical on both disks that make up the MD device.

This section describes how to create an MD device. This is the only step that you must complete before you enable Software RAID for a package. The other RAID operations are needed only during maintenance or during recovery process after a failure has occurred.

NOTE For all the steps in the subsequent sections, all the persistent device names, and not the actual device names, must be used for the two component disks of the MD mirror.

To Create and Assemble an MD Device

This example shows how to create the MD device `/dev/md0`, you must create it from a LUN of storage device 1 (`/dev/hpdev/sde1`) and another LUN from storage 2 (`/dev/hpdev/sdf1`).

Run the following command to create an MD device:

```
# mdadm --create --verbose /dev/md0 --level=1 \
--raid-devices=2 /dev/hpdev/sde1 /dev/hpdev/sdf1
```

This command creates the MD device.

Once the new RAID device, `/dev/md0`, is created on one of the cluster nodes, you must assemble it on the nodes where the package must run. You create an MD device only once and you can manage other functions using the XDC scripts.

To assemble the MD device, complete the following procedure:

1. Stop the MD device on the node where you created it, by running the following command:

   ```
   # mdadm -S /dev/md0
   ```

2. Assemble the MD device on the other node by running the following command:

   ```
   # mdadm -A -R /dev/md0 /dev/hpdev/sde1 /dev/hpdev/sdf1
   ```

3. Stop the MD device on the other node by running the following command:

   ```
   # mdadm -S /dev/md0
   ```

   You must stop the MD device soon after you assemble it on the second node.

4. If you want to create volume groups, restart the MD device on the first node by running the following command:

   ```
   # mdadm -A -R /dev/md0 /dev/hpdev/sde1 /dev/hpdev/sdf1
   ```

5. After you have created the volume groups, stop the MD device by running the following command:

   ```
   # mdadm -S /dev/md0
   ```

IMPORTANT You would need to repeat this procedure to create all MD devices that are used in a package.

When data is written to this device, the MD driver writes to both the underlying disks. In case of read requests, the MD reads from one device or the other based on its algorithms. After creating this device you treat it like any other LUN that is going to have shared data in a Serviceguard environment and then create a logical volume and a file system on it.

Creating Volume Groups and Configuring VG Exclusive Activation on the MD Mirror

Once you create the MD mirror device, you need to create volume groups and logical volumes on it.

NOTE

XDC A.01.00 does not support configuring multiple raid1 devices as physical volumes in a single volume group.

For example, if you create a volume group vg01, it can have only one MD raid1 device `/dev/md0` as its physical volume.

To configure multiple raid1 devices as physical volumes in a single volume group, you must install the XDC A.01.02 patch. To install this patch, you must first upgrade to HP Serviceguard A.11.18 and the latest version of XDC. After upgrading, install the A.01.02 patch that is specific to the operating system in your environment.

XDC A.01.02 contains the following patches for Red Hat and SuSE Linux operating systems:

- SGLX_00133 for Red Hat Enterprise Linux 4
- SGLX_00134 for Red Hat Enterprise Linux 5
- SGLX_00135 for SUSE Linux Enterprise Server 10

You can contact the HP support personnel to obtain these patches.

When you create a logical volume on an MD device, the actual physical devices that form the MD raid1 mirror must be filtered out to avoid receiving messages from LVM about duplicate PV entries.

For example, let us assume that `/dev/sde` and `/dev/sdf` are two physical disks that form the md device `/dev/md0`. The persistent device names for `/dev/sde` and `/dev/sdf` are `/dev/hpdev/md0_mirror0` and `/dev/hpdev/md0_mirror1` respectively. When you create a logical volume, duplicate entries are detected for the two physical disks that form the mirror device. As a result, the logical volume is not created and an error message is displayed. Following is a sample of the error message that is displayed:

```
Found duplicate PV 9w3TIxKZ6lFRqWUmQm9tlV5nsdUkTi4i: using
/dev/sde not /dev/sdf
```

With this error, you cannot create a new volume group on `/dev/md0`. As a result, you must create a filter for LVM. To create a filter, add the following line in the `/etc/lvm/lvm.conf` file:

```
filter = [ "r|/dev/cdrom|","r|/dev/hpdev/md0_mirror0|",
"r|/dev/hpdev/md0_mirror1|" ]
```

where `/dev/hpdev/md0_mirror0` and `/dev/hpdev/md0_mirror1` are the persistent device names of the devices `/dev/sde` and `/dev/sdf` respectively.

NOTE When adding the filter, ensure that you use the persistent names of all the devices used in the mirrors.

This prevents these mirror devices from being scanned or used for logical volumes. You have to reload LVM with `/etc/init.d/lvm force-reload`.

Once you add the filter to the `/etc/lvm/lvm.conf` file, create the logical volume infrastructure on the MD mirror device as you would on a single disk. For more information on creating volume groups and logical volumes, see the latest edition of the *Managing HP Serviceguard 11.16 for Linux* at **http://docs.hp.com/en/ha.html#Serviceguard for Linux**

Configuring the Package Control Script and RAID Configuration File

This section describes the package control scripts and configuration files that you need to create and edit to enable Software RAID in your Serviceguard environment.

Earlier versions of Serviceguard supported MD as a multipathing software. As a result, the package control script includes certain configuration parameters that are specific to MD. Do not use these parameters to configure XDC in your environment. Following are the parameters in the configuration file that you must not edit:

```
# MD (RAID) CONFIGURATION FILE
# Specify the configuration file that will be used to define
# the md raid devices for this package.
# NOTE: The multipath mechanisms that are supported for shared storage
# depend on the storage subsystem and the HBA driver in the
# configuration. Follow the documentation for those devices when setting
# up multipath. The MD driver was used with earlier versions of
# Serviceguard and may still be used by some storage system/HBA
# combinations. For that reason there are references to MD in the template
# files, worksheets, and other areas. Only use MD if your storage systems
# specifically calls out its use for multipath.
# If some other multipath mechanism is used (e.g. one built
# into an HBA driver), then references to MD, RAIDTAB, RAIDSTART, etc.
# should be commented out. If the references are in the comments, they
# can be ignored. References to MD devices, such as /dev/md0, should be
# replaced with the appropriate multipath device name.
# For example:
# RAIDTAB="/usr/local/cmcluster/conf/raidtab.sg"
#RAIDTAB=""
# MD (RAID) COMMANDS
```

```
# Specify the method of activation and deactivation for md.
# Leave the default (RAIDSTART="raidstart", "RAIDSTOP="raidstop") if you want
# md to be started and stopped with default methods.
RAIDSTART="raidstart -c ${RAIDTAB}"
RAIDSTOP="raidstop -c ${RAIDTAB}"
```

Creating and Editing the Package Control Scripts

After you install the XDC software, you need to create a package control script and add references to the XDC software to enable Software RAID. After you create the package control script you need to complete the following tasks:

- Edit the value of the DATA_REP variable
- Edit the value of the XDC_CONFIG_FILE to point to the location where the raid.conf file is placed
- Configure the RAID monitoring service

To Create a Package Control Script

The procedure to create a package control script for XDC software is identical to the procedure that you follow to create other package control scripts.

To create a package control script, run the following command:

 # cmmakepkg –s <package file name>.sh

For example: **# cmmakepkg –s oracle_pkg.sh**

An empty template file for this package is created. You will need to edit this package control script, in order to enable Software RAID in your environment.

To Edit the DATA_REP Variable

The DATA_REP variable defines the nature of data replication that is used. To enable Software RAID, set the value of this variable to MD. You must set this value for every package that you need to enable Software RAID. When you set this parameter to xdcmd, it enables remote data replication through Software RAID.

For example: **DATA_REP="xdcmd"**

To Edit the XDC_CONFIG FILE parameter

In addition to modifying the DATA_REP variable, you must also set XDC_CONFIG_FILE to specify the raid.conf file for this package. This file resides in the package directory.

For example: **XDC_CONFIG_FILE="$SGCONF/oracle_pkg/raid.conf"**

To Configure the RAID Monitoring Service

After you have edited the variables in the XDC configuration file (XDC_CONFIG_FILE), you must set up RAID monitoring as a service within Serviceguard. Following is an example of how the file content must look:

```
SERVICE_NAME[0]="RAID_monitor"
SERVICE_CMD[0]="$SGSBIN/raid_monitor '${XDC_CONFIG_FILE}'"
SERVICE_RESTART[0]=""
```

Ensure that this service is also configured in the package configuration file as shown below:

```
SERVICE_NAME raid_monitor
SERVICE_FAIL_FAST_ENABLED YES
SERVICE_HALT_TIMEOUT 300
```

After editing the package control script, you must edit the raid.conf file to enable Software RAID.

Editing the raid.conf File

The raid.conf file specifies the configuration information of the RAID environment of the package. You must place a copy of this file in the package directory of every package that you have enabled Software RAID. The parameters in this file are:

- RPO_TARGET

  Given a set of storage that is mirrored remotely, as in Figure 1-4, the RPO_TARGET (Recovery Point Objective Target) is the maximum time allowed between the expiration of the Link_Down_Timeout (t1 in Figure 3-1, after the failure of the data links to the remote storage) and the package starting up on the remote node (t4 on Figure 3-1). If

more time elapses than what is specified for RPO_TARGET, the package is prevented from starting on the remote node (assuming that the node still has access only to its own half of the mirror).

By default, RPO_TARGET is set to 0. Leave it at 0 to ensure the package does not start on an adoptive node with a mirror half that is not current. This ensures the highest degree of data currency.

If RPO_TARGET is not set to 0, the value of RAID_MONITOR_INTERVAL should be less than the value of RPO_TARGET. (RAID_MONITOR_INTERVAL should also be less than the value of the Link_Down_Timeout parameter so that disk access failure can be recognized early enough in certain failure scenarios.)

IMPORTANT

A very low value of RAID_MONITOR_INTERVAL (less than 5 seconds) has some impact on system performance because of the high frequency of polling.

You can also set RPO_TARGET to the special value -1 or to any positive integer. Setting RPO_TARGET to -1 causes the RAID system to ignore any time-window checks on the disk set. This allows the package to start with a mirror half that is not current.

Setting the RPO_TARGET to any positive integer, means that the package will start with a mirror half that is not current by any number of seconds less than that value. For example, an RPO_TARGET of 45 means that the package will start only if the mirror is up to date, or out of date by less than 45 seconds.

Because some timers are affected by polling, the value of this parameter can vary by approximately 2 seconds.

This also requires that the minimum value of this parameter is 2 seconds if a small value is necessary. Change the value of RPO_TARGET, if necessary, after considering the cases discussed below.

Cases to Consider when Setting RPO_TARGET

RPO_TARGET allows for certain failure conditions when data is not synchronized between the two sites.

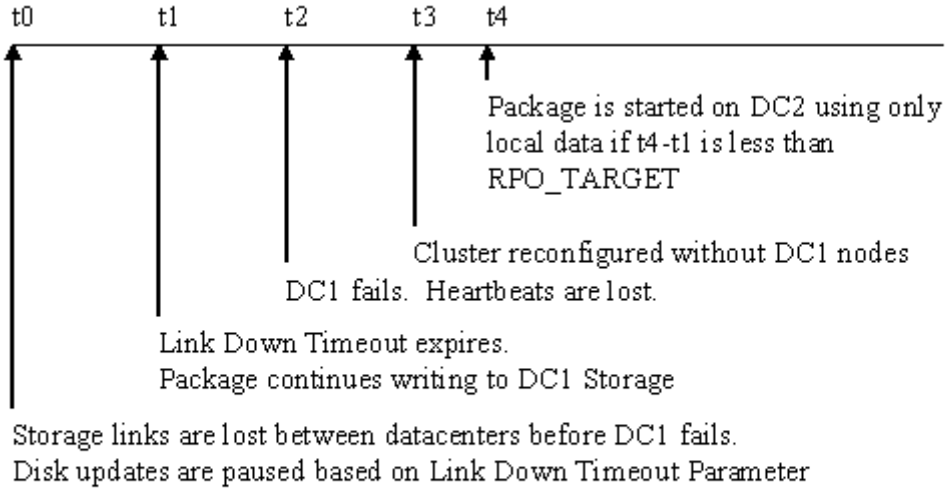
For example, let us assume that the data storage links in Figure 1-4 fail before the heartbeat links fail. In this case, after the time specified by `Link_Down_Timeout` has elapsed, a package in Datacenter1 (DC1) will continue updating the local storage, but not the mirrored data in datacenter2 (DC2). While the communication links must be designed to prevent this situation as far as possible, this scenario could occur and may last for a while before one of the sites fails.

NOTE

For more information on how access to disks is disabled in certain failure scenarios, see “Setting the Value of the Link Down Timeout Parameter” on page 69.

Let us consider a few failure scenarios and the impact of different `RPO_TARGET` values. The discussion below is based on the timeline and events shown in Figure 3-1.

Figure 3-1 RPO Target Definitions



To ensure that no data is lost when a package fails over to DC2 and starts with only DC2's local storage, t2 must occur between t0 and t1.

Now consider an XDC configuration such as that shown in Figure 1-3 (DWDM links between data centers). If DC1 fails such that links A and B both fail simultaneously, and DC1's connection to the Quorum Server fails at the same time, Serviceguard ensures that DC2 survives and the package fails over and runs with DC2 local storage.

But if DC1's links A and B fail, and later DC1's link to the Quorum Server fails, then both sets of nodes (DC1 and DC2) will try to obtain the cluster lock from the Quorum Server. If the Quorum server chooses DC1 (which is about to experience complete site failure), then the entire cluster will go down.

But if the Quorum Server chooses DC2 instead, then the application running on DC1 will not be able to write to the remote storage but will continue to write to its local (DC1) storage until site failure occurs (at t3). If the network is set up in such a way that the application cannot communicate with its clients under these circumstances, the clients will not receive any acknowledgement of these writes. HP recommends you configure the network such that when links between the sites fail, the communication links to the application clients also go down.

If the network is configured to prevent the application from communicating with its clients under these circumstances, the clients will not receive any acknowledgement of these writes and after the failover will re-transmit them, and the writes will be committed and acknowledged at DC2. This is the desired outcome; HP recommends you configure the network such that when links between the sites fail, the communication links to the application clients are also shut down.

In the case of an XDC configuration such as that shown in Figure 1-4, there is an additional variable in the possible failure scenarios. Instead of a DWDM link, in this configuration there are two separate LAN and FC links which can experience failure independent of each other. If the network links between the sites fail within a very short period (on the order of 1 second) after t1 (after the storage links had failed), the XDC software on DC1 will not have time to inform the XDC on DC2 of the failure. So DC2 assumes that there were no updates after t1, but there may have been.

When this scenario occurs, disk writes continue on DC1 until t3. In this case, the effective value of the `RPO_TARGET` parameter is greater than the expected value of `0`.

Again, if the network is set up in such a way that when the links between the sites fail, the communication links to the application clients are also shut down, then the unintended writes are not acknowledged and have no long term effect.

IMPORTANT

The value you set for RPO_TARGET must be more than the value you set for the RAID_MONITOR_INTERVAL parameter. By default, the RAID_MONITOR_INTERVAL parameter is set to 30 seconds.

For example: RPO_TARGET=60 seconds

- MULTIPLE_DEVICES AND COMPONENT_DEVICES

  Parameter RAID_DEVICE [ ] specifies the MD devices that are used by a package. You must begin with RAID_DEVICE[0], and increment the list in sequence. Component device parameters DEVICE_0[ ] and DEVICE_1[ ] specify the component devices for the MD device of the same index.

  For example, if a package uses multiple devices such as

  — md0 consisting of devices /dev/hpdev/sde and /dev/hpdev/sdf and

  — md1 consisting of devices /dev/hpdev/sdg1 and /dev/hpdev/mylink-sdh1

  use

```
# md0
RAID_DEVICE[0]=/dev/md0;
DEVICE_0[0]="/dev/hpdev/sde";
DEVICE_1[0]="/dev/hpdev/sdf"
#md1
RAID_DEVICE[1]=/dev/md1;
DEVICE_0[1]="/dev/hpdev/sdg1";
DEVICE_1[1]="/dev/hpdev/sdh1"
```

  The MD RAID device names and the component device names must be unique across the packages in the entire cluster.

- RAID_MONITOR_INTERVAL

  This parameter defines the time interval, in seconds, the raid monitor script waits between each check to verify accessibility of both component devices of all mirror devices used by this package. By default, this parameter is set to 30 seconds.

IMPORTANT After you edit the parameters, ensure that you copy the package control script and the edited `raid.conf` file to all nodes in the cluster. All the nodes in the cluster must have the identical copy of the files.

After you have installed the XDC software, and completed the configuration procedures, your environment is equipped to handle any failure scenarios. The subsequent chapter discusses how certain disaster scenarios are handled by the XDC software.

4 Disaster Scenarios and Their Handling

The previous chapters provided information on deploying Software RAID in your environment. In this chapter, you will find information on how Software RAID addresses various disaster scenarios. All the disaster scenarios described in this section have the following three categories:

- **Disaster Scenario**

  Describes the type of disaster and provides details regarding the cause and the sequence of failures leading to the disasters in the case of multiple failures.

- **What happens when this disaster occurs**

  Describes how the Extended Distance Cluster software handles this disaster.

- **Recovery Process**

  After the disaster strikes and necessary actions are taken by the software to handle it, you need to ensure that your environment recovers from the disaster. This section describes all the steps that an administrator needs to take to repair the failures and restore the cluster to its original state. Also, all the commands listed in this column must be executed on a single line.

The following table lists all the disaster scenarios that are handled by the Extended Distance Cluster software. All the scenarios assume that the setup is the same as the one described in "Extended Distance Clusters" on page 18 of this document.

Table 4-1 Disaster Scenarios and Their Handling

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
A package (P1) is running on a node (Node 1). Node 1 experiences a failure.	The package (P1) fails over to another node (Node 2). This node (Node 2) is configured to take over the package when it fails on Node 1.	As the network and both the mirrored disk sets are accessible on Node 2, and were also accessible when Node 1 failed, you only need to restore Node 1. Then you must enable the package to run on Node 1 after it is repaired by running the following command: `# cmmodpkg -e P1 -n N1`

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
A package (P1) is running on a node (Node 1). The package uses a mirror (md0) that consists of two storage components - S1 (local to Node 1 - /dev/hpdev/mylink-sde) and S2 (local to Node 2). Access to S1 is lost from both nodes, either due to power failure to S1 or loss of FC links to S1.	The package (P1) continues to run on Node 1 with the mirror that consists of only S2.	Once you restore power to S1, or restore the FC links to S1, the corresponding mirror half of S1 (/dev/hpdev/mylink-sde) is accessible from Node 1. To make the restored mirrored half part of the MD array, complete the following procedure: 1. Run the following command to remove the mirrored half from the array: **# mdadm --remove /dev/md0 /dev/hpdev/mylink-sde** 2. Run the following command to add the mirrored half to the array: **# mdadm --add /dev/md0 /dev/hpdev/mylink-sde** The re-mirroring process is initiated. When it is complete, the extended distance cluster detects the added mirror half and accepts S1 as part of md0.

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
A package (P1) is running on a node (Node 1). The package uses a mirror (md0) that consists of two storage components - S1 (local to Node 1 - /dev/hpdev/mylink-sde) and S2 (local to Node 2) Data center 1 that consists of Node 1 and P1 experiences a failure. **NOTE:** In this example, failures in a data center are instantaneous. For example - power failure.	The package (P1) fails over to Node 2 and starts running with the mirror of md0 that consists of only the storage local to node 2 (S2).	Complete the following procedure to initiate a recovery: 1. Restore data center 1, Node 1 and storage 1. Once Node 1 is restored, it rejoins the cluster. Once S1 is restored, it becomes accessible from Node 2. When the package failed over and started on Node 2, S1 was not a part of md0. As a result, you need to add S1 into md0. Run the following command to add S1 to md0: **# mdadm – -add /dev/md0 /dev/hpdev/mylink-sde** The re-mirroring process is initiated. When it is complete, the extended distance cluster detects the added mirror half and accepts S1 as part of md0. 2. Enable P1 to run on Node 1 by running the following command: **# cmmodpkg -e P1 -n N1**

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
This is a multiple failure scenario where the failures occur in a particular sequence in the configuration that corresponds to figure 2 where Ethernet and FC links do not go over DWDM. The package (P1) is running on a node (N1). P1 uses a mirror md0 consisting of S1 (local to node N1, say `/dev/hpdev/mylink-sde`) and S2 (local to node N2). The first failure occurs with all FC links between the two data centers failing, causing N1 to lose access to S2 and N2 to lose access to S1. After recovery for the first failure has been initiated, the second failure occurs when re-mirroring is in progress and N1 goes down.	The package (P1) continues to run on N1 after the first failure, with md0 consisting of only S1. After the second failure, the package (P1) fails over to N2 and starts with S1. Since S2 is also accessible, the extended distance cluster adds S2 and starts re-mirroring of S2.	For the first failure scenario, complete the following procedure to initiate a recovery: 1. Restore the links in both directions between the data centers. As a result, S2 (`/dev/hpdev/mylink-sdf`) is accessible from N1 and S1 is accessible from N2. 2. Run the following commands to remove and add S2 to md0 on N1: **`# mdadm --remove /dev/md0 /dev/hpdev/mylink-sdf`** **`# mdadm --add /dev/md0 /dev/hpdev/mylink-sdf`** The re-mirroring process is initiated. The re-mirroring process starts from the beginning on N2 after the second failure. When it completes, the extended distance cluster detects S2 and accepts it as part of `md0` again.

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
This is a multiple failure scenario where the failures occur in a particular sequence in the configuration that corresponds to figure 2 where Ethernet and FC links do not go over DWDM. The RPO_TARGET for the package P1 is set to IGNORE. The package is running on Node 1. P1 uses a mirror md0 consisting of S1 (local to node N1, - /dev/hpdev/mylink-sde) and S2 (local to node N2). The first failure occurs when all FC links between the two data centers fail, causing Node 1 to lose access to S2 and Node 2 to lose access to S1. After sometime a second failure occurs. Node 1 fails (because of power failure)	The package (P1) continues to run on Node 1 after the first failure, with the MD0 that consists of only S1. After the second failure, the package P1 fails over to N2 and starts with S2. Data that was written to S1 after the FC link failure is now lost because the RPO_TARGET was set to IGNORE.	In this scenario, no attempts are made to repair the first failure until the second failure occurs. Typically the second failure occurs before the first failure is repaired. 1. To recover from the first failure, restore the FC links between the data centers. As a result, S1 is accessible from N2. 2. Run the following command to add S1 to md0 on N2: **# mdadm --add /dev/md0 /dev/hpdev/mylink-sde** This command initiates the re-mirroring process. When it is complete, the extended distance cluster detects S1 and accepts it as md0. For the second failure, restore N1. Once it is restored, it joins the cluster and can access S1 and S2. 1. Run the following command to enable P1 to run on N1 **# cmmodpkg -e P1 -n N1**

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
This failure is the same as the previous failure except that the package (P1) is configured with RPO_TARGET set to 60 seconds. In this case, initially the package (P1) is running on N 1. P1 uses a mirror md0 consisting of S1 (local to node N1 - /dev/hpdev/mylink-sde) and S2 (local to node N2). The first failure occurs when all FC links between the two data centers fail, causing N1 to lose access to S2 and N2 to lose access to S1. After the package resumes activity and runs for 20 seconds, a second failure occurs causing N 1 to fail, perhaps due to power failure.	Package P1 continues to run on N1 after the first failure with md0 consisting of only S1 After the second failure, package P1 fails over to N2 and starts with S2. This happens because the disk S2 is non-current by less than 60 seconds. This time limit is set by the RPO_TARGET parameter. Disk S2 has data that is older than the other mirror half S1. However, all data that was written to S1 after the FC link failure is lost	In this scenario, no attempts are made to repair the first failure until the second failure occurs. Typically, the second failure occurs before the first failure is repaired. 1. To recover from the first failure, restore the FC links between the data centers. As a result, S1 (/dev/hpdev/mylink-sde) is accessible from N2. 2. Run the following command to add S1 to md0 on N2: **# mdadm --add /dev/md0 /dev/hpdev/mylink-sde**. This command initiates the re-mirroring process. When it is complete, the extended distance cluster detects S1 and accepts it as md0 again. For the second failure, restore N1. Once it is restored, it joins the cluster and can access S1 and S2. 1. Run the following command to enable P1 to run on N1 **# cmmodpkg -e P1 -n N1**

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
In this case, the package (P1) runs with RPO_TARGET set to 60 seconds. Package P1 is running on node N1. P1 uses a mirror md0 consisting of S1 (local to node N1, for example /dev/hpdev/mylink-sde) and S2 (local to node N2). The first failure occurs when all FC links between two data centers fail, causing N1 to lose access to S2 and N2 to lose access to S1. After the package resumes activity and runs for 90 seconds, a second failure occurs causing node N1 to fail.	The package (P1) continues to run on N1 with md0 consisting of only S1 after the first failure After the second failure, the package does not start up on N2 because when it tries to start with only S2 on N2, it detects that S2 is non-current for a time period which is greater than the value of RPO_TARGET.	In this scenario, no attempts are made to repair the first failure until the second failure occurs. Complete the following procedure to initiate a recovery: 1. To recover from the first failure, restore the FC links between the data centers. As a result, S1 is accessible from N2. 2. After the FC links are restored, and S1 is accessible from N2, run the following command to restart the package on N2. **# cmrunpkg <package_name>** When the package starts up on N2, it automatically adds S1 back into the array and starts re-mirroring from S1 to S2. When re-mirroring is complete, the extended distance cluster detects and accepts S1 as part of md0 again. For the second failure, restore N1. Once it is restored, it joins the cluster and can access S1 and S2. 1. Run the following command to enable P1 to run on N1: **# cmmodpkg -e P1 -n N1**

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
This scenario is an extension of the previous failure scenario. In the previous scenario, when the package fails over to N2, it does not start as the value of RPO_TARGET would have been exceeded. To forcefully start the package P1 on N2 when the FC links are not restored on N2, check the package log file on N2 and execute the commands that appear in it.	If the FC links are not restored on N2, you can only start the package forcefully. You can forcefully start a package only if it is determined that the associated data loss is acceptable. After you execute the force start commands, package P1 starts on N2 and runs with md0 consisting of only S2 (/dev/hpdev/mylink-sdf).	Complete the following procedure to initiate a recovery: 1. Reconnect the FC links between the data centers. As a result, S1 (/dev/hpdev/mylink-sde) becomes accessible from N2 2. Run the following command to add S1 to md0 on N2 **# mdadm --add /dev/md0 /dev/hpdev/mylink-sde** This command initiates the re-mirroring process from S2 to S1. When re-mirroring is complete, the extended distance cluster detects S1 and accepts it as part of md0.

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
In this case, the package (P1) runs with RPO-TARGET set to 60 seconds. In this case, initially the package (P1) is running on node N1. P1 uses a mirror md0 consisting of S1 (local to node N1, for example /dev/hpdev/mylink-sde) and S2 (local to node N2). The first failure occurs when all FC links between the two data centers fail, causing N1 to lose access to S2 and N2 to lose access to S1. Immediately afterwards, a second failure occurs where node (N1) goes down because of a power failure. After N1 is repaired and brought back into the cluster, package switching of P1 to N1 is enabled. **IMPORTANT:** While it is not a good idea to enable package switching of P1 to N1, it is described here to show recovery from an operator error. The FC links between the data centers are not repaired and N2 becomes inaccessible because of a power failure.	When the first failure occurs, the package (P1) continues to run on N1 with md0 consisting of only S1. When the second failure occurs, the package fails over to N2 and starts with S2. When N2 fails, the package does not start on node N1 because a package is allowed to start only once with a single disk. You must repair this failure and both disks must be synchronized and be a part of the MD array before another failure of same pattern occurs. In this failure scenario, only S1 is available to P1 on N1, as the FC links between the data centers are not repaired. As P1 started once with S2 on N2, it cannot start on N1 until both disks are available.	Complete the following steps to initiate a recovery: 1. Restore the FC links between the data centers. As a result, S2 (/dev/hpdev/mylink-sdf) becomes available to N1 and S1 (/dev/hpdev/mylink-sde) becomes accessible from N2. 2. To start the package P1 on N1, check the package log file in the package directory and run the commands which will appear to force a package start. When the package starts up on N1, it automatically adds S2 back into the array and the re-mirroring process is started. When re-mirroring is complete, the extended distance cluster detects and accepts S1 as part of md0.

Table 4-1 Disaster Scenarios and Their Handling (Continued)

Disaster Scenario	What Happens When This Disaster Occurs	Recovery Process
In this case, initially the package (P1) is running on node N1. P1 uses a mirror md0 consisting of S1 (local to node N1, for example /dev/hpdev/mylink-sde) and S2 (local to node N2). The first failure occurs with all Ethernet links between the two data centers failing.	With this failure, the heartbeat exchange is lost between N1 and N2. This results in both nodes trying to get to the Quorum server. If N1 accesses the Quorum server first, the package continues to run on N1 with S1 and S2 while N2 is rebooted. If N2 accesses the Quorum server, the package fails over to N2 and starts running with both S1 and S2 and N1 is rebooted.	Complete the following steps to initiate a recovery: 1. You need to only restore the Ethernet links between the data centers so that N1 and N2 can exchange heartbeats 2. After restoring the links, you must add the node that was rebooted as part of the cluster. Run the cmrunnode command to add the node to the cluster. **NOTE:** If this failure is a precursor to a site failure, and if the Quorum Service arbitration selects the site that is likely to have a failure, it is possible that the entire cluster will go down.
In this case, initially the package (P1) is running on node N1. P1 uses a mirror md0 consisting of S1 (local to node N1, say /dev/hpdev/mylink-sde) and S2 (local to node N2). The first failure occurs when the Ethernet links from N1 to the Ethernet switch in datacenter1 fails.	With this failure, the heartbeat exchange between N1 and N2 is lost. N2 accesses the Quorum server, as it is the only node which has access to the Quorum server. The package fails over to N2 and starts running with both S1 and S2 while N1 gets rebooted.	Complete the following procedure to initiate a recovery: 1. Restore the Ethernet links from N1 to the switch in data center 1. 2. After restoring the links, you must add the node that was rebooted as part of the cluster. Run the cmrunnode command to add the node to the cluster.

A Managing an MD Device

This chapter includes additional information on how to manage the MD device. For the latest information on how to manage and MD device, see *The Software-RAID HOWTO* manual available at:
http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html

Following are the topics discussed in this chapter:

Viewing the Status of the MD Device

After creating an MD device, you can view its status. By doing so, you can remain informed of whether the device is clean, up and running, or if there are any errors.

To view the status of the MD device, run the following command on any node:

`cat /proc/mdstat`

Immediately after the MD devices are created and during some recovery processes, the devices undergo a re-mirroring process. You can view the progress of this process in the `/proc/mdstat` file. Following is the output you will see:

```
[root@dlhct1 ~]# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sde[2] sdf[0]
9766784 blocks [2/1] [U_]
[=>..................] recovery = 8.9% (871232/9766784) finish=2.7min
speed=54452K/sec
unused devices: <none>
```

NOTE A status report obtained using the **`cat /proc/mdstat`** command shows the MD device name and the actual device names of the two MD component devices. It does not show the persistent device names.

After you create an MD device, you can view the status of the device, stop and start the device, add and remove a mirror component from the MD device.

Stopping the MD Device

After you create an MD device, it begins to run. You need to stop the device and add the configuration into the `raid.conf` file. To stop the MD device, run the following command:

`# mdadm -S <md_device name>`

When you stop this device, all resources that were previously occupied by this device are released. Also, the entry of this device is removed from the `/proc/mdstat` file.

Example A-1 Stopping the MD Device /dev/md0

To stop the MD device `/dev/md0`, run the following command:

`[root@dlhct1 dev]# mdadm -S /dev/md0`

Once you stop the device, the entry is removed from the `/proc/mdstat` file. Following is an example of what the file contents will look like:

```
[root@dlhct1 dev]# cat /proc/mdstat
Personalities : [raid1]
unused devices: <none>
```

NOTE This command and the other commands described subsequently are listed as they may be used during cluster development and during some recovery operations.

Starting the MD Device

After you create an MD device, you would need to stop and start the MD device to ensure that it is active. You would not need to start the MD device in any other scenario as this is handled by the XDC software.

To start the MD device, run the following command:

```
# mdadm -A -R <md_device_name>
<md_mirror_component_persistent_name_0>
<md_mirror_component_persistent_name_1>
```

Example A-2 Starting the MD Device /dev/md0

To start the MD device /dev/md0, run the following command:

```
# mdadm -A -R /dev/md0 /dev/hpdev/sde /dev/hpdev/sdf1
```

Following is an example of what the /proc/mdstat file contents will look like once the MD is started:

```
[root@dlhct1 dev]# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sde[1] sdf[0]
9766784 blocks [2/2] [UU]
unused devices: <none>
```

Removing and Adding an MD Mirror Component Disk

There are certain failure scenarios, where you would need to manually remove the mirror component of an MD device and add it again later. For example, if links between two data centers fail, you would need to remove and add the disks that were marked as failed disks.

When a disk within an MD device fails, the `/proc/mdstat` file of the MD array displays a message. For example:

```
[root@dlhct1 dev]# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sde[2](F) sdf[0]
9766784 blocks [2/1] [U_]
unused devices: <none>
```

In the message, the `(F)` indicates which disk has failed. In this example, the `sde[2]` disk has failed.

In such a scenario, you must remove the failed disk from the MD array. You need to determine the persistent name of the failed disk before you remove it from the MD array. For this example, run the following command to determine the persistent name of the disk:

`# udevinfo -q symlink -n sdc1`

Following is a sample output:

```
hpdev/mylink-sdc \
disk/by-id/scsi-3600805f3000b9510a6d7f8a6cdb70054-part1 \
disk/by -path/pci-0000:06:01.0-scsi-0:0:1:30-part1
```

Run the following command to remove a failed component device from the MD array:

`# mdadm - -remove <md device name>`
`<md_mirror_component_persistent_name>`

In this example:

`# mdadm --remove /dev/md0 /dev/hpdev/mylink-sdc1`

This command removes the failed mirrored disk from the array.

Example A-3 **Removing a failed MD component disk from /dev/md0 array**

To remove a failed MD component disk from /dev/md0, run the following command:

 # mdadm –-remove /dev/md0 /dev/hpdev/sde

Following is an example of the status message that is displayed when a failed component is removed from the MD array:

```
[root@dlhct1 dev]# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdf[0]
9766784 blocks [2/1] [U_]
unused devices: <none>
```

Adding a Mirror Component Device

As mentioned earlier, in certain failure scenarios, you need to remove a failed mirror disk component, repair it and then add it back into an MD array. Run the following command to add a mirror component back into the MD array:

 # mdadm - -add <md device name> <md_mirror_component_persistent_name>

Example A-4 **Adding a new disk as an MD component to /dev/md0 array**

To add a new disk to the /dev/md0 array, run the following command:

 # mdadm - -add /dev/md0 /dev/hpdev/sde

Following is an example of the status message displayed in the /proc/mdstat file once the disk is added:

```
Personalities : [raid1]
md0 : active raid1 sde[2] sdf[0]
9766784 blocks [2/1] [U_]
[=>..................] recovery = 8.9% (871232/9766784) finish=2.7min
speed=54452K/sec
unused devices: <none>
```