

AIX Version 7.3

Device management



Note

Before using this information and the product it supports, read the information in [“Notices” on page 239](#).

This edition applies to AIX Version 7.3 and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright International Business Machines Corporation 2021, 2024.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this document.....	vii
Highlighting.....	vii
Case-sensitivity in AIX.....	vii
ISO 9000.....	vii
 Device management.....	 1
What's new.....	1
Logical Volume Manager.....	1
Logical Volume Manager concepts.....	1
Configuring Logical Volume Manager.....	10
Troubleshooting LVM.....	27
Logical volume storage.....	37
Preparing to install a device.....	38
Configuring a read/write optical drive.....	39
Configuration of a large number of devices.....	39
Adding a removable media drive.....	40
Space Reclamation Support for Logical Volume Storage.....	41
Logical volume storage concepts.....	41
Configuring Logical Volume Storage.....	46
Volume group strategy.....	54
Logical volume strategy.....	55
Implementing a volume group policy.....	64
Paging space and virtual memory.....	65
Paging space concepts.....	65
Configuring paging space.....	68
Troubleshooting paging space.....	71
Virtual Memory Manager.....	72
File systems.....	73
File system concepts.....	74
Configuring file systems.....	82
Managing file system.....	82
Maintaining file systems.....	85
Troubleshooting file systems.....	94
Disk overflows.....	96
Mounting.....	100
File system types.....	105
Directories.....	119
Workload manager.....	126
Workload management concepts.....	127
Administering Workload Manager.....	133
Classes.....	144
Process classifications in Workload Manager.....	148
Resource management with Workload Manager.....	152
Setting up Workload Manager.....	159
Troubleshooting Workload Manager.....	161
Workload Manager API.....	161
Examples of Workload Manager classification, rules, and limits.....	164
Workload Manager commands.....	166
Device nodes.....	166
Device classes.....	167

Device configuration database and device management.....	167
Device states.....	168
Device location codes.....	168
Adapter location codes.....	168
Printer and plotter location codes.....	169
tty location codes.....	169
SCSI device location codes.....	170
Dials/LPFKeys location codes.....	170
Multiprotocol port location codes.....	171
Device drivers.....	171
Setting up an iSCSI offload adapter.....	172
Configuring the iSCSI adapter in AIX.....	172
Updating the flat file of an iSCSI target.....	173
Adding a statically-discovered iSCSI target into ODM.....	173
Adding statically-discovered iSCSI targets from a flat file into ODM	173
PCI hot plug management.....	174
Displaying PCI hot-plug slot information.....	175
Unconfiguring PCI communications adapters.....	176
Removing or replacing a PCI hot plug adapter.....	176
Adding a PCI hot plug adapter.....	177
Multiple Path I/O.....	177
MPIO-capable device management.....	178
Configuring an MPIO device.....	181
Encrypting physical volumes.....	181
Supported multi-path devices.....	182
MPIO device attributes.....	183
Path control module attributes.....	185
SAN replication attributes.....	187
Communications adapter removal.....	189
Unconfiguring storage adapters.....	194
Unconfiguring async adapters.....	195
Troubleshooting I/O devices.....	195
Targeted device configuration.....	199
Targeted configuration of FC and FCoE devices.....	199
Tape drives.....	200
Tape drive attributes.....	200
Special files for tape drives.....	210
USB device support.....	212
USB flash drive support.....	212
USB Blu-ray drive read-only support.....	213
AIX USB device quirks.....	214
Caching storage data.....	215
Concept.....	215
Advantages.....	215
Limitations.....	216
Components.....	217
Configuring.....	217
Managing.....	222
Monitoring cache statistics.....	223
Login names, system IDs, and passwords.....	224
Logging in to the operating system.....	224
Logging in more than one time (login command).....	225
Becoming another user on a system (su command).....	225
Suppressing login messages.....	226
Logging out of the operating system (exit and logout commands).....	226
Displaying user IDs.....	226
Passwords.....	228
Command summary for login names, system IDs, and passwords.....	230

Common Desktop Environment.....	230
Enabling and disabling desktop autostart.....	230
Starting the Common Desktop Environment manually.....	231
Stopping the Common Desktop Environment manually.....	231
Modifying desktop profile.....	231
Adding and removing displays and terminals for the Common Desktop Environment.....	231
Displaying device customization for the Common Desktop Environment.....	233
Live Partition Mobility with Host Ethernet Adapters.....	235
Requirements for Live Partition Mobility with HEA	236
Running Live Partition Mobility with HEA.....	236
Relocating an adapter for DLPAR.....	238
Loopback device.....	238
Notices.....	239
Privacy policy considerations.....	240
Trademarks.....	241
Index.....	243

About this document

This document provides users and system administrators with complete information that can affect your selection of options when performing such tasks as backing up and restoring the system, managing physical and logical storage, sizing appropriate paging space, and so on. It provides complete information about how to perform such tasks as managing logical volumes, storage, and resources. System users can learn how to perform such tasks as running commands, handling processes, handling files and directories, and basic printing.

Other topics useful to users and system administrators include creating and resizing paging space, managing virtual memory, backing up and restoring the system, managing hardware and pseudodevices, using the System Resource Controller (SRC), securing files, using storage media, customizing environment files, and writing shell scripts. This document is also available on the documentation CD that is shipped with the operating system.

Highlighting

The following highlighting conventions are used in this document:

Bold	Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.
<i>Italics</i>	Identifies parameters whose actual names or values are to be supplied by the user.
Monospace	Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.

Case-sensitivity in AIX

Everything in the AIX operating system is case-sensitive, which means that it distinguishes between uppercase and lowercase letters. For example, you can use the **ls** command to list files. If you type **LS**, the system responds that the command is not found. Likewise, **FILEA**, **FiLea**, and **filea** are three distinct file names, even if they reside in the same directory. To avoid causing undesirable actions to be performed, always ensure that you use the correct case.

ISO 9000

ISO 9000 registered quality systems were used in the development and manufacturing of this product.

Device management



You can use commands to manage the different devices that are available in AIX. Some of the devices that you can manage include Logical Volume Manager, file systems, tape drives, and printers.

What's new in Device management

Read about new or significantly changed information for the Device management topic collection.

How to see what's new or changed

To help you see where technical changes have been made, the content uses:

- The  image to mark where new or changed information begins.
- The  image to mark where new or changed information ends.

November 2022

The following is a summary of the updates that are made to this topic collection:

- Added information about [AIX USB device quirks](#) under the [USB device support](#) topic.
- Added [Encrypting physical volumes](#) information under [Multiple Path I/O](#)

December 2021

The following information is a summary of the updates made to this topic collection:

- Updated information about the attributes in the [“Path control module attributes” on page 185](#) topic.

Logical Volume Manager

The set of operating system commands, library subroutines, and other tools that allow you to establish and control logical volume storage is called the Logical Volume Manager (LVM).

The LVM controls disk resources by mapping data between a more simple and flexible *logical* view of storage space and the actual *physical* disks. The LVM does this using a layer of device-driver code that runs above traditional disk device drivers.

The LVM consists of the logical volume device driver (LVDD) and the LVM subroutine interface library. The *logical volume device driver* (LVDD) is a pseudo-device driver that manages and processes all I/O. It translates logical addresses into physical addresses and sends I/O requests to specific device drivers. The *LVM subroutine interface library* contains routines that are used by the system management commands to perform system management tasks for the logical and physical volumes of a system.

Logical Volume Manager concepts

Before you can start using Logical Volume Manager you must understand the basic mechanics and terminology.

Vary-on process

The vary-on process is one of the mechanisms that the LVM uses to ensure that a volume group is ready to use and contains the most up-to-date data.

The **varyonvg** and **varyoffvg** commands activate or deactivate (make available or unavailable for use) a volume group that you have defined to the system. The volume group must be varied on before the system can access it. During the vary-on process, the LVM reads management data from the physical

volumes defined in the volume group. This management data, which includes a volume group descriptor area (VGDA) and a volume group status area (VGSA), is stored on each physical volume of the volume group.

The VGDA contains information that describes the mapping of physical partitions to logical partitions for each logical volume in the volume group, as well as other vital information, including a time stamp. The VGSA contains information such as which physical partitions are stale and which physical volumes are missing (that is, not available or active) when a vary-on operation is attempted on a volume group.

If the vary-on operation cannot access one or more of the physical volumes defined in the volume group, the command displays the names of all physical volumes defined for that volume group and their status. This helps you decide whether to vary-off this volume group.

Quorum

The quorum is one of the mechanisms that the LVM uses to ensure that a volume group is ready to use and contains the most up-to-date data.

A quorum is a vote of the number of Volume Group Descriptor Areas and Volume Group Status Areas (VGDA/VGSA) that are active. A quorum ensures data integrity of the VGDA/VGSA areas in the event of a disk failure. Each physical disk in a volume group has at least one VGDA/VGSA. When a volume group is created onto a single disk, it initially has two VGDA/VGSA areas residing on the disk. If a volume group consists of two disks, one disk still has two VGDA/VGSA areas, but the other disk has one VGDA/VGSA. When the volume group is made up of three or more disks, then each disk is allocated just one VGDA/VGSA.

A quorum is lost when at least half of the disks (meaning their VGDA/VGSA areas) are unreadable by LVM. In a two-disk volume group, if the disk with only one VGDA/VGSA is lost, a quorum still exists because two of the three VGDA/VGSA areas still are reachable. If the disk with two VGDA/VGSA areas is lost, this statement is no longer true. The more disks that make up a volume group, the lower the chances of quorum being lost when one disk fails.

When a quorum is lost, the volume group varies itself off so that the disks are no longer accessible by the LVM. This prevents further disk I/O to that volume group so that data is not lost or assumed to be written when physical problems occur. Additionally, as a result of the vary-off, the user is notified in the error log that a hardware error has occurred and service must be performed.

There are cases when it is desirable to continue operating the volume group even though a quorum is lost. In these cases, quorum checking can be turned off for the volume group. This type of volume group is referred to as a *nonquorum volume group*. The most common case for a nonquorum volume group occurs when the logical volumes have been mirrored. When a disk is lost, the data is not lost if a copy of the logical volume resides on a disk that is not disabled and can be accessed. However, there can be instances in nonquorum volume groups, mirrored or nonmirrored, when the data (including copies) resides on the disk or disks that have become unavailable. In those instances, the data might not be accessible even though the volume group continues to be varied on.

Mirror Pools

Mirror pools make it possible to divide the physical volumes of a volume group into separate pools.

A mirror pool is made up of one or more physical volumes. Each physical volume can only belong to one mirror pool at a time. When creating a logical volume, each copy of the logical volume being created can be assigned to a mirror pool. Logical volume copies that are assigned to a mirror pool will only allocate partitions from the physical volumes in that mirror pool. This provides the ability to restrict the disks that a logical volume copy can use. Without mirror pools, the only way to restrict which physical volume is used for allocation when creating or extending a logical volume is to use a map file. Thus, using mirror pools greatly simplify this process. Mirror pools can be created with the **extendvg** command or the **chpv** command.

You must specify a mirror pool name when you create a new mirror pool. Mirror pool names must conform to the following rules:

- Can only contain alphanumeric characters or the _ (underscore), - (minus sign), or . (period) characters.
- Must be less than or equal to 15 characters.
- Must be unique in the volume group.

Once mirror pools have been used in a volume group, the volume group can no longer be imported into a version of AIX that does not support mirror pools. This includes any version of AIX before 6.1.1.0. Additionally, in order to use mirror pools with enhanced concurrent-mode LVM, all nodes in the cluster must support mirror pools.

Mirror pool strictness

Mirror pool strictness can be used to enforce tighter restrictions on mirror pool use. Mirror pool strictness can have one of the following three values:

off

When mirror pool strictness is set to **off**, no restrictions are placed on mirror pool use. This is the default value.

on

When mirror pool strictness is set to **on**, each logical volume copy created in the volume group must be assigned to a mirror pool.

super

When mirror pool strictness is set to **super**, the following restrictions apply:

- Local and remote physical volumes cannot belong to the same mirror pool.

Note: For more information on local and remote physical volumes, refer to the HACMP/XD GLVM documentation.

- There can be a maximum of three mirror pools in a volume group.
- Each mirror pool must contain at least one copy of each logical volume in the volume group.

Geographic Logical Volume Manager

Geographic Logical Volume Manager (GLVM) allows you to maintain a mirror copy of your data at a geographically distant location.

GLVM can help protect your business from a disaster by mirroring critical data to a remote disaster recovery site. If a disaster, such as a fire or flood, were to destroy the data at your production site, you would have a backup copy of the data at your disaster recovery site.

The data is mirrored over standard TCP/IP networks. The production and disaster recovery sites need not be on the same physical network. Routers and gateways can be used between the two sites. Instead of extremely long disk cables, the TCP/IP network and the Remote Physical Volume (RPV) device driver are used for remote disk access.

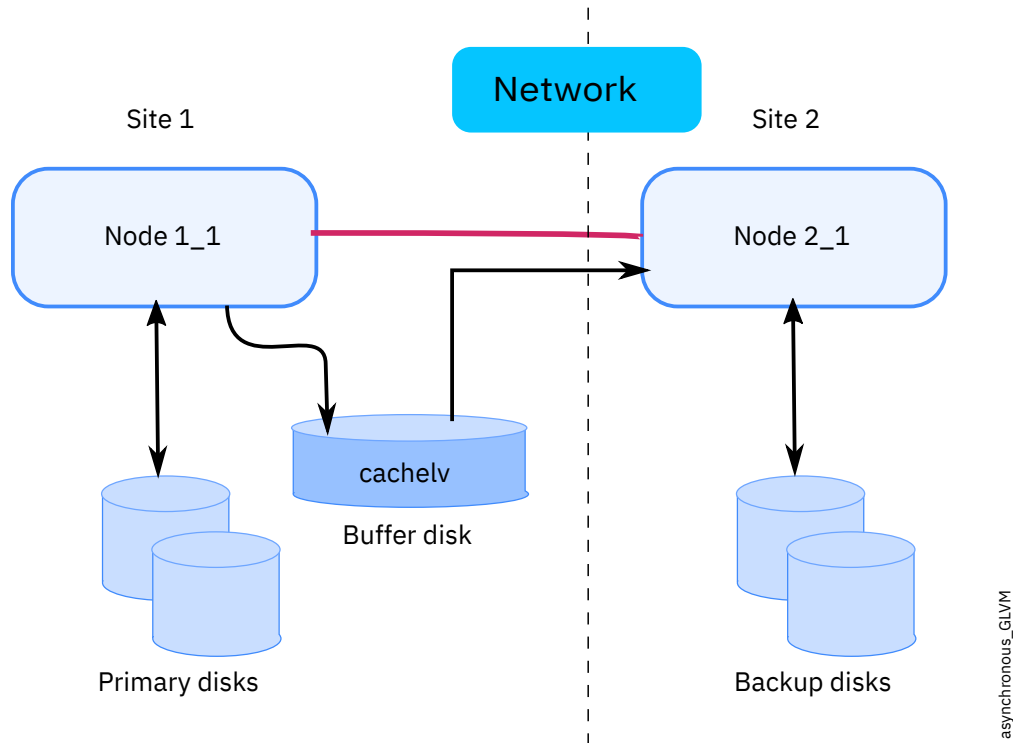
The user configures geographically distant disks as remote physical volumes and then combines those remote physical volumes with local physical volumes to form geographically mirrored volume groups. These volume groups are managed by Logical Volume Manager (LVM) and work similar to the generic volume groups. GLVM supports both synchronous and asynchronous remote mirroring.

Asynchronous GLVM

Asynchronous Geographic Logical Volume Manager (GLVM) mirrors data across the primary and secondary sites over standard TCP/IP networks. Where the primary site can be your production site and the secondary site can be your disaster recovery site.

The following figure shows the asynchronous GLVM operations, where the GLVM environment contains two nodes, Node 1_1 and Node 2_1. These nodes are linked across Site 1 and Site 2 through the standard TCP/IP network.

Figure 1.



The workloads that are running in Node 1_1 of the primary site (Site 1) writes data to the local disks (primary disks) that are in the primary site. These write operations are tracked by the Logical Volume Manager (LVM) and the workloads that are running in Node 2_1 of the secondary site (Site 2) writes data to the cache logical volume (cache) which is a buffer disk in the primary site. The asynchronous GLVM transports the data from the cache logical volume, to the remote site (Site 2) through the TCP/IP network and writes the data to the backup disks in the remote site. The workloads that are running in Node 2_1 of the remote site also writes data to the backup disks in the secondary site.

Planning for asynchronous GLVM

You must consider optimal network speed, size of the cache logical volume (cachelv) and so on, when you are deploying GLVM for asynchronous mirroring.

To deploy GLVM for asynchronous remote mirroring, consider the following guidelines:

- Ensure the required optimal network speed depending on the data size in your environment:

Data size	Network speed
Less than 1 TB	1 Gbps or higher
1 - 10 TB	5 Gbps or higher
10 TB - 100 TB	10 Gbps or higher

- Ensure that the size of the cache logical volume (cachelv) in the primary site is allocated appropriately so that the primary site contains sufficient disk space for the cachelv logical volume. Calculate the maximum disk space for the buffer disk that is required in the primary site based on the peak I/O operations and network bandwidth. Allocate double the amount of buffer disk space to the cachelv logical volume.
- Monitor the usage of CPU and memory resources in the environment by using performance tools after a week of GLVM deployment to identify the peak periods and the amount of I/O operations. Ensure that CPU and memory usage does not exceed more than 80% of its overall capacity even during peak periods. An operating system failure might occur when the logical partition (LPAR) memory is exhausted.

- Ensure that the LVM and GLVM tunable parameters are not modified across sites while deploying GLVM for asynchronous remote mirroring. If you want to modify the tunable parameters, bring the GLVM offline by using the **varyoffvg** command and modify the LVM or GLVM configuration settings. For example, if you want to change the `max_transfer_size` tunable parameter by using the **chdev** command, ensure that the value of the `max_transfer_size` tunable parameter is same across sites while deploying GLVM for asynchronous remote mirroring. Otherwise, I/O errors might result across sites.
- You cannot perform split volume group operations using GLVM that supports asynchronous mirroring.
- GLVM that supports asynchronous mirroring supports only non-concurrent scalable volume groups. GLVM that supports asynchronous mirroring does not support any other volume groups, such as snapshot volume groups.
- When a node in the primary site fails, the primary site attempts to retain the workloads on the recovery site. If the GLVM that supports asynchronous mirroring is brought online, the time taken to bring the GLVM online depends on the residual data size of workloads that is pending in the `cache1v` logical volume. The residual data must be transferred across the network to the recovery site. During this period, write operations that are performed by the application cannot complete until the cache recovery operation is complete. Thus, plan for some downtime during the cache recovery operation to ensure the recovery synchronization of the residual data.
- GLVM that supports asynchronous mirroring provides many tunable parameters that you can modify based on the requirements of your environment.
 - Review and modify the `RPVlevel` tunable parameter based on your network delays. You can use the `RPVlevel` tunable parameter to specify the default timeout duration that is allowed to complete synchronization operation. The default time out value is 180 seconds.
 - Increase the number of memory (physical) buffer disks that are assigned to LVM to manage the `cache1v` logical volume. You can use 16,000, which is an optimal value.
- Some of the LVM metadata related operations require synchronous I/O operations across sites to ensure that the LVM metadata is correct on both sites. You can perform these types of synchronous I/O operations only when previously buffered data in the `cache1v` logical volume is transferred completely to the recovery site. Hence, these type of operations might take a long time while waiting for the buffered data to get transferred to the target site. If you need faster operations, plan to perform the synchronous I/O operations when the residual buffer data in the `cache1v` logical volume is minimal. You can use the **rpvstat -C** command to check the residual buffer data in the `cache1v` disks. The following operations might also take time to complete because of the residual buffer data:
 - Reduction of logical volume size or reduction of file system size.
 - Removal of logical volume.
 - Closing the GLVM that supports asynchronous mirroring.
- Synchronization of the remote partition might fail if the residual buffer data is large in the `cache1v` logical volume. The synchronization might require more time to complete than the default value of the `RPVlevel` tunable parameter. For successful synchronization operation, update the `RPVlevel` tunable parameter depending on the residual buffer data.
- When GLVM is configured with more than 900 disks on an LPAR, increase value of the DMA setting for FC adapter by running the following command:

```
# chdev -l fcs1 -a lg_term_dma=0x80000000 -P
# rmdev -Rl fcs1
# cfgmgr -l fcs1 -v
```

- Asynchronous GLVM supports maximum 1020 number of rpvclients per LPAR, if the one network is configured per rpvclient device.

Table 1. Maximum number of rpvclients supported	
Networks used for each rpvclient	Maximum number of rpvclients supported
1	1020

Table 1. Maximum number of rpvclients supported (continued)	
Networks used for each rpvclient	Maximum number of rpvclients supported
2	510
3	340
4	255

Setting up asynchronous GLVM

By default the Geographic Logical Volume Manager (GLVM) packages are installed in the AIX® operating system.

Prerequisites

- Ensure that your system is running on AIX version AIX 7 with 7200-04, or later.
- Ensure that you are using TCP/UDP port 6192 across the primary and secondary sites for communication.
- Ensure that you have sufficient disks for creating the remote physical volume (RPV) server and RPV clients.

To set up the asynchronous GLVM, complete the following steps:

1. Configure the RPV server and RPV clients by using the SMIT glvm_utils fast path, as follows:
 - a. To create an RPV server, select **Remote Physical Volume Servers > Add Remote Physical Volume Servers**, and then press Enter.
For more information about creating the RPV server and RPV clients in PowerHA® SystemMirror® for AIX, see [Configuring geographically mirrored volume groups](#).
 - b. To define an RPV site name, select **Remote Physical Volume Servers > Remote Physical Volume Server Site Name Configuration > Define / Change / Show Remote Physical Volume Server Site Name**, and then press Enter.
 - c. To create RPV client, select **Remote Physical Volume Clients > Add Remote Physical Volume Clients**, and then press Enter
2. Create an asynchronous volume group that can be used with the GLVM, by completing the following steps:
 - a. Create a scalable volume group by running the following command:

```
mkvg -f -S -y agmvg hdisk5 hdisk15
```

When you run this command, the asynchronous agmvg volume group is created. The volume group has two disks, *hdisk5* and *hdisk15*.

- b. Create a mirror pool for the disks that are associated with the asynchronous volume group by running the following command:

```
chpv -p mp1 hdisk5 hdisk15
```

When you run this command, a mirror pool mp1 is created for the *hdisk5* and *hdisk15* disks.

- i) Now, extend the asynchronous volume group to include other disks that might be created when you configure the RPV server and RPV clients in the node of the primary site.

```
extendvg -f -p mp2 agmvg hdisk22 hdisk23
```

When you run this command, the agmvg asynchronous volume group is extended to include *hdisk22* and *hdisk23* disks. The mirror pool mp2 is also created.

- c. Create a logical volume (glv) for the agmvg asynchronous volume group by running the following command. The file system of the logical volume consists of files that are part of data logical volume and log logical volume.

- Create data logical volume by running the following command:

```
mklv -t jfs2 -y glv -p copy1=mp1 -b n -s s -u 1 agmvg 10
```

- Create log logical volume by running the following command:

```
mklv -t jfs2log -y glv_log -p copy1=mp1 -b n -s s -u 1 agmvg 10
```

- d. Create a journaled file system (JFS2) file system that must be used for the data logical volume and log logical volume by running the following command:

```
crfs -v jfs2 -A no -m /gfs -d glv -a logname=glv_log
```

- e. Create a mirror copy of the asynchronous volume group in the RPV server and the RPV client.

```
mirrorvg -c 2 -p copy2=mp2 agmvg
```

All data from mirror pool mp1 (disks in the primary site) is copied to the mirror pool mp2 (remote disks in the secondary site) in synchronous mode over the standard TCP/IP network.

- f. Create cache logical volumes for caching asynchronous volume group write requests by running the following commands.

```
mklv -t aio_cache -y mp1_cache -p copy1=mp1 -b n agmvg 5 hdisk5  
mklv -t aio_cache -y mp2_cache -p copy1=mp2 -b n agmvg 5 hdisk23
```

These cache logical volumes must not be mirrored across sites. You can create mirror pools by using the super strict disk allocation policy that can manage the new aio_cache logical volumes.

- g. Configure the asynchronous mirroring properties of the cache logical volumes and the secondary site to specify the maximum capacity for write requests before new write requests must wait for mirroring to the remote disks.

- Configure asynchronous mirroring properties of the cache LV by running the following command:

```
chmp -A -h 80 -m mp2 agmvg
```

- Configure asynchronous mirroring properties of the secondary site by running the following command:

```
chmp -A -h 50 -m mp1 agmvg
```

- Verify whether the **chmp** command is successful by running the following command:

```
lsmp agmvg
```

After you configure asynchronous mirroring properties, if the state of the mirror pool reaches the asynchronous state and the mirror pool is active, remote I/O operation requests are written to the cache logical volume (cache1v) and then the I/O operation requests are mirrored from the cache1v to the secondary site. You can then mount the file system of the logical volume in the secondary site to start the application that is mirrored from primary site to secondary site.

3. Verify the asynchronous GLVM setup by performing the following steps:

- Ensure proper network connectivity and ensure that the state of the physical volume (PV) disk is active by running the following command:

```
# lsvg -p agmvg
```

An output that is similar to the following example is displayed:

```
agmpvg:
PV_NAME  PV_STATE TOTAL_PPs FREE_PPs FREE DISTRIBUTION
hdisk6   active   957      627      192..00..52..191..192
hdisk1   active   957      937      192..171..191..191..192
hdisk8   active   951      617      192..00..42..191..192
hdisk9   active   951      947      192..181..191..191..192
```

- Monitor the asynchronous I/O operation requests and the number of connections that are active in the GLVM environment, by running the following command:

```
rpvstat -A
```

An output that is similar to the following example is displayed:

```
Remote Physical Volume Statistics:
RPV Client ax Completed Async KB Writes Completed Async KB Writes Cached Async Writes Cached Async KB Writes Pending Async Writes Pending Async KB Writes
-----
hdisk9      A           0           0           0           0           0           0
hdisk8      A    2061696    1018176796        21       10240           0           0
```

You can use the **rpvstat -n** command to check the status and usage of network that connects the sites. You can use the **rpvstat -A** command to check the status of the asynchronous I/O operation. You can use the **rpvstat -C** command to check statistics of the cache logical volume.

Best practices for asynchronous GLVM deployment

Consider the following best practices for asynchronous Graphical Logical Volume Manager (GLVM) deployment.

- Configure the value of the `RPV level I/O timeout` parameter by using the **chdev** command to avoid any network latency problems that are related to the I/O timeout value. You can modify this timeout parameter when the `rpv` disk is in the defined state. The default value of this parameter is 180 seconds.
- In a stand-alone GLVM environment, you must ensure that all the backup disks in the secondary sites are in an active state before you bring the volume group online. During the online recovery of the volume group, if the RPV device driver detects that the RPV server is not online to perform a specific asynchronous I/O operation, the RPV device driver updates in the cache disk about the failed request. The I/O operations continue running synchronously. If you want to revert to asynchronous mirroring after you rectify the problem, you can use the **chmp** command.
- After a site fails, the state of asynchronous mirroring in secondary site might be inactive. After you integrate the secondary site again with the primary site, the mirror pool must be converted to asynchronous mirroring to continue asynchronous mirroring deployment.
- Monitor regularly whether the asynchronous mirroring state of the GLVM is active by using the **lsmp** command. The **rpvstat -A** command shows the statistics and states of asynchronous mirroring.
- Monitor the asynchronous I/O operations and number of active connections in the GLVM environment regularly. The **rpvstat -n** command displays details of about a specific network. The **rpvstat -A** command displays details about asynchronous I/O operations. The **rpvstat -C** command displays statistics about the cache logical volumes.
- For better performance, ensure that the disk driver parameters of the storage device that is deployed in your environment is configured correctly.

Nonquorum volume groups

The Logical Volume Manager (LVM) automatically deactivates the volume group when it lacks a quorum of Volume Group Descriptor Areas (VGDA) or Volume Group Status Areas (VGSAs). However, you can

choose an option that allows the group to stay online as long as there is one VGDA/VGSA pair intact. This option produces a *nonquorum volume group*.

The LVM requires access to all of the disks in nonquorum volume groups before allowing reactivation. This ensures that the VGDA and VGSA are up-to-date.

You might want to produce a nonquorum volume group in systems where every logical volume has at least two copies. If a disk failure occurs, the volume group remains active as long as there is one active disk.

Note: Both user-defined and **rootvg** volume groups can operate in nonquorum status, but the methods used to configure user-defined volume groups and **rootvg** volume groups as nonquorum and for recovery after hardware failures are different. Be sure you use the correct method for the appropriate volume group.

Even when you are using nonquorum volume groups, it is possible to lose quorum and see the following message in the **errpt** command output:

```
QUORUM LOST, VOLUME GROUP CLOSING LVM.
```

This message occurs when all physical volumes are in the missing state and the LVM automatically varies off the volume group.

The message says QUORUM LOST because disabling quorum on a volume group reduces the quorum requirement to 1. You can use the **lsvg vgname** command to display the quorum value which is in the QUORUM: field. In the case where all physical volumes are missing, even this minimum quorum requirement is violated, resulting in the lost quorum message and an automatic vary off of the volume group.

Conversion of a volume group to nonquorum status

You can change a volume group to nonquorum status to have data continuously available even when there is no quorum.

This procedure is often used for systems with the following configurations:

- A two-disk volume group in which the logical volumes are mirrored
- A three-disk volume group in which the logical volumes are mirrored either once or twice

When a volume group under these circumstances can operate in nonquorum status, then even if a disk failure occurs, the volume group remains active as long as at least one disk in the volume group is active.

To make recovery of nonquorum groups possible, ensure the following:

- If your system uses JFS or JFS2 file systems, mirror the JFS log logical volume.
- Place mirrored copies on separate disks. If you are unsure of the configuration, type the following command to check the physical location (PV1, PV2, and PV3) of each logical partition. (To place the copies on separate disks, the PV1, PV2, and PV3 columns must contain different hdisk numbers.)

```
lslv -m LVName
```

If a logical volume has its only copies residing on the same disk, and that disk becomes unavailable, the volume will not be available to the user regardless of the quorum or nonquorum status of its volume group.

Both user-defined and rootvg volume groups can operate in nonquorum status, but their configuration and recovery methods are different.

To activate a nonquorum user-defined volume group, all of the volume group's physical volumes must be accessible or the activation fails. Because nonquorum volume groups stay online until the last disk becomes inaccessible, it is necessary to have each disk accessible at activation time.

Attention: When a disk associated with the rootvg volume group is missing, avoid powering on the system unless the missing disk cannot possibly be repaired. The Logical Volume Manager (LVM) always uses the -f flag to forcibly activate (vary on) a nonquorum rootvg; this operation involves risk. LVM must force the

activation because the operating system cannot be started unless rootvg is activated. In other words, LVM makes a final attempt to activate (vary on) a nonquorum rootvg even if only a single disk is accessible.

Related concepts

[High availability in case of adapter or power supply failure](#)

To protect against adapter or power supply failure, depending on your requirements, do one or more of the following.

Configuring Logical Volume Manager

The Logical Volume Manager (LVM) is installed with the base operating system and needs no further configuration. However, disks must be configured and defined as a physical volume before the LVM can use them.

Related tasks

[Defining a raw logical volume for an application](#)

A *raw logical volume* is an area of physical and logical disk space that is under the direct control of an application, such as a database or a partition, rather than under the direct control of the operating system or a file system.

LVM maintenance commands and fastpaths

The simplest tasks you might need when maintaining the entities that LVM controls (physical and logical volumes, volume groups, and file systems) are grouped within the following table.

Table 2. Managing Logical Volumes and Storage Tasks		
Task	SMIT Fast Path	Command or File
Activate a volume group	smit varyonvg	
Add a fixed disk without data to existing volume group	smit extendvg	
Add a fixed disk without data to new volume group	smit mkvg	
Add a logical volume ^{Note 1}	smit mklv	
Add a volume group	smit mkvg	
Add and activate a new volume group	smit mkvg	
Change a logical volume to use data allocation	smit chlv1	
Change the name of a volume group ^{Note 2}	1. smit varyoffvg 2. smit exportvg 3. smit importvg 4. smit mountfs	1. varyoffvg <i>OldVGName</i> 2. exportvg <i>OldVGName</i> 3. importvg <i>NewVGName</i> 4. mount all
Change a volume group to use automatic activation	smit chvg	
Change or set logical volume policies	smit chlv1	
Copy a logical volume to a new logical volume ^{Note 3}	smit cplv	

Table 2. Managing Logical Volumes and Storage Tasks (continued)

Task	SMIT Fast Path	Command or File
Copy a logical volume to an existing logical volume of the same size ^{Attn 1}	smit cplv	
Copy a logical volume to an existing logical volume of smaller size ^{Attn 1 Note 3}	Do not use SMIT ^{Attn 2}	<ol style="list-style-type: none"> 1. Create logical volume. For example: mklv -y hdiskN vg00 4 2. Create new file system on new logical volume. For example: crfs -v jfs -d hdiskN -m /doc -A yes 3. Mount file system. For example: mount /doc 4. Create directory at new mount point. For example: mkdir /doc/options 5. Transfer files system from source to destination logical volume. For example: cp -R /usr/adam/oldoptions/* /doc/options
Copy a logical volume to an existing logical volume of larger size ^{Attn 1}	smit cplv	
Deactivate a volume group	smit varyoffvg	
Enable write-verify and change scheduling policy	smit chlvs	
Increase the maximum size of a logical volume	smit chlvs	
Increase the size of a logical volume	smit extendlv	
List all logical volumes by volume group	smit lslvs	
List all physical volumes in system	smit lspv	
List all volume groups	smit lsvg	
List the status, logical volumes, or partitions of a physical volume	smit lspv	
List the contents of a volume group	smit lsvg1	
List a logical volume's status or mapping	smit lslv	
Mirror a logical volume with or without data allocation	smit mklvcopy	

Table 2. Managing Logical Volumes and Storage Tasks (continued)

Task	SMIT Fast Path	Command or File
Power off a removable disk	smit offdisk	Available with the hot-removability feature only
Power on a removable disk	smit ondisk	Available with the hot-removability feature only
Remove mirroring from a volume group	smit unmirrorvg	
Remove a volume group	smit reducevg2	
Reorganize a volume group	smit reorgvg	
Unconfigure and power off a disk	smit rmdisk1 or smit rmdisk then smit opendoor	



Attention:

1. Using this procedure to copy to an existing logical volume will overwrite any data on that volume without requesting user confirmation.
2. Do not use the SMIT procedure or the **cp1v** command to copy a larger logical volume to a smaller one. Doing so results in a corrupted file system because some of the data (including the superblock) is not copied to the smaller logical volume.

Note:

1. After you create a logical volume, the state will be closed because no LVM structure is using that logical volume. It will remain closed until a file system has been mounted over the logical volume or the logical volume is opened for raw I/O.
2. You cannot change the name of, import, or export **rootvg**.
3. You must have enough direct access storage to duplicate a specific logical volume.

Adding disks while the system remains available

The following procedure describes how to turn on and configure a disk using the hot-removability feature, which lets you add disks without powering off the system.

You can add a disk for additional storage or to correct a disk failure. This feature is only available on certain systems.

1. Install the disk in a free slot of the cabinet. For detailed information about the installation procedure, see the service guide for your machine.
2. Power on the new disk by typing the following fast path on the command line:

```
smit ondisk
```

At this point, the disk is added to the system but it is not yet usable. What you do next depends on whether the new disk contains data.

- If the disk has no data, add it as a physical volume to a volume group using one of the following:
 - To add the disk to an existing volume group, type the following fast path on the command line:

```
smit extendvg
```

- To add the disk to a new volume group, type the following fast path on the command line:

```
smit mkvg
```

- If the disk contains data, import the data.

Changing the name of a logical volume

The following procedure describes how to rename a logical volume without losing data on the logical volume.

In the following examples, the logical volume name is changed from `lv00` to `lv33`.

1. Unmount all file systems associated with the logical volume, by typing:

```
umount /FSname
```

Where *FSname* is the full name of a file system.

Note:

- a. The **umount** command fails if the file system you are trying to unmount is currently being used. The **umount** command executes only if none of the file system's files are open and no user's current directory is on that device.
 - b. Another name for the **umount** command is **umount**. The names are interchangeable.
2. Rename the logical volume, by typing:

```
chlv -n NewLVname OldLVname
```

Where the **-n** flag specifies the new logical volume name (*NewLVname*) and *OldLVname* is the name you want to change. For example:

```
chlv -n lv33 lv00
```

Note: If you rename a JFS or JFS2 log, the system prompts you to run the **chfs** command on all file systems that use the renamed log device.

3. Remount the file systems you unmounted in step “1” on page 13 by typing:

```
mount /test1
```

At this point, the logical volume is renamed and available for use.

Copying a logical volume to another physical volume

Depending on your needs, there are several ways to copy a logical volume to another physical volume while retaining file system integrity.

There are multiple methods for copying a logical volume or JFS to another physical volume. Choose the method that best serves your purposes.

Copying a logical volume

The simplest method is to use the **cp1v** command to copy the original logical volume and create a new logical volume on the destination physical volume.

1. Stop using the logical volume. Unmount the file system, if applicable, and stop any application that accesses the logical volume.
2. Select a physical volume that has the capacity to contain all of the data in the original logical volume.



Attention: If you copy from a larger logical volume containing data to a smaller one, you can corrupt your file system because some data (including the superblock) might be lost.

3. Copy the original logical volume (in this example, it is named **lv00**) and create the new one, using the following command:

Note: The following **cp1v** command fails if it creates a new logical volume and the volume group is varied on in concurrent mode.

```
cp1v lv00
```

4. Mount the file systems, if applicable, and restart applications to begin using the logical volume.

At this point, the logical volume copy is usable.

Copying a logical volume while original logical volume remains usable

If your environment requires continued use of the original logical volume, you can use the **splitlvcopy** command to copy the contents, as shown in the following example.

1. Mirror the logical volume, using the following SMIT fast path:

```
smit mklvcopy
```

2. Stop using the logical volume. Unmount the file system, if applicable, and stop or put into quiescent mode any application that accesses the logical volume.



Attention: The next step uses the **splitlvcopy** command. Always close logical volumes before splitting them and unmount any contained file systems before using this command. Splitting an open logical volume can corrupt your file systems and cause you to lose consistency between the original logical volume and the copy if the logical volume is accessed simultaneously by multiple processes.

3. With root authority, copy the original logical volume (**oldlv**) to the new logical volume (**newlv**) using the following command:

```
splitlvcopy -y newlv oldlv
```

The **-y** flag designates the new logical volume name. If the **oldlv** volume does not have a logical volume control block, the **splitlvcopy** command completes successfully but generates a message that the **newlv** volume has been created without a logical volume control block.

4. Mount the file systems, if applicable, and restart applications to begin using the logical volume.

At this point, the logical volume copy is usable.

Copying a raw logical volume to another physical volume

To copy a raw logical volume to another physical volume, perform the following steps.

1. Create a mirrored copy of the logical volume on a new physical volume in the volume group using the following command:

```
mklvcopy LogVol_name 2 new_PhysVol_name
```

2. Synchronize the partitions in the new mirror copy using the following command:

```
syncvg -l LogVol_name
```

3. Remove the copy of the logical volume from the physical volume using the following command:

```
rmlvcopy LogVol_name 1 old_PhysVol_name
```

At this point, the raw logical volume copy is usable.

Creating a file system log on a dedicated disk for a user-defined volume group

A JFS or JFS2 file system log is a formatted list of file system transaction records. The log ensures file system integrity (but not necessarily data integrity) in case the system goes down before the transactions are completed.

A dedicated disk is created on hd8 for rootvg when the system is installed. The following procedure helps you create a JFS log on a separate disk for other volume groups. When you create a JFS2 log, the procedure requires the following changes:

- The log device type is **jfs2log**.
- The **logform** command requires the **-V jfs2** option to specify a JFS2 log device.
- The **crfs** commands must specify **jfs2** instead of **jfs**.

Note: There is no requirement for a JFS2 log to be on a separate disk as the file system. The only requirement is that the log devices must be on the same volume group as the file system. In this procedure, the JFS2 log must be on a separate disk for performance improvement.

Creating a file system log file for user-defined volume groups can improve the performance under certain conditions, such as, if you have an NFS server and you want the transactions for this server to be processed without competition from other processes.

You can use the following procedure, which creates a volume group (fsvg1) with two physical volumes (hdisk1 and hdisk2). The file system is on hdisk2 (a 256 MB file system that is mounted at /u/myfs) and the log is on hdisk1. By default, a JFS log size is 4 MB. You can place little-used programs, for example, /blv, on the same physical volume as the log without affecting performance.

To create a JFS log for a user-defined volume group by using the SMIT and the command-line interface, follow these steps:

1. Add the new volume group (in this example, fsvg1) by using the SMIT fast path:

```
smit mkvg
```

2. Add a new logical volume to this volume group by using the SMIT fast path:

```
smit mklv
```

3. On the **Add a Logical Volume** screen, add your data to the following fields. For example:

Logical Volumes NAME	fsvg1log
Number of LOGICAL PARTITIONS	1
PHYSICAL VOLUME names	hdisk1
Logical volume TYPE	jfslog
POSITION on Physical Volume	center

4. After you set the fields, press Enter to accept your changes and exit SMIT.
5. Type the following command on a command line:

```
/usr/sbin/logform /dev/fsvg1log
```

6. When you receive the following prompt, type **y** and press Enter:

```
Destroy /dev/fsvg1log
```

Despite the wording in this prompt, nothing is destroyed. When you respond y to this prompt, the system formats the logical volume for the JFS log so that it can record file system transactions.

7. Add another logical volume by using the following SMIT fast path:

```
smit mklv
```

8. Type the name of the same volume group as you used in step 2 (fsvg1 in this example). In the Logical Volumes screen, add your data to the following fields. Remember to designate a different physical volume for this logical volume than you did in step 3. For example:

Logical Volumes NAME	fs1v1
Number of LOGICAL PARTITIONS	64
PHYSICAL VOLUME names	hdisk2
Logical volume TYPE	jfs

After you set the fields, press Enter to accept your changes and exit SMIT.

9. Add a file system to the new logical volume, designate the log, and mount the new file system, by using the following sequence of commands:

```
crfs -v jfs -d LogVolName -m FileSysName -a logname=FSLogPath
mount FileSysName
```

Where *LogVolName* is the name of the logical volume you created in step 2; *FileSysName* is the name of the file system you want to mount on this logical volume; and *FSLogPath* is the name of the logical volume you created in step 2. For example:

```
crfs -v jfs -d fslv1 -m /u/myfs -a logname=/dev/fsvg1log
mount /u/myfs
```

10. To verify that you set up the file system and log correctly, type the following command (substituting your volume group name) :

```
lsvg -l fsvg1
```

The output shows both logical volumes that you created, with their file system types, as in the following example:

```
LV NAME          TYPE  ...
/dev/fsvg1log    jfslog ...
fslv1            jfs   ...
```

You created a volume group that contains at least two logical volumes on separate physical volumes, and one of those logical volumes contains the file system log.

Note: To provide redundancy, you can provide mirroring on the logical volume level for the JFS2 log device. However, providing mirroring is not a common practice and is not necessary.

Importing or exporting a volume group

The following table explains how to use import and export to move a user-defined volume group from one system to another. (The rootvg volume group cannot be exported or imported.)

The export procedure removes the definition of a volume group from a system. The import procedure serves to introduce the volume group to its new system.

You can also use the import procedure to reintroduce a volume group to the system when it once was associated with the system but had been exported. You can also use import and export to add a physical volume that contains data to a volume group by putting the disk to be added in its own volume group.



Attention: The **importvg** command changes the name of an imported logical volume if a logical volume of that name already exists on the new system. If the **importvg** command must rename a logical volume, it prints an error message to standard error. When there are no conflicts, the **importvg** command also creates file mount points and entries in the `/etc/filesystems` file.

Import and Export Volume Group Tasks		
Task	SMIT Fast Path	Command or File
Import a volume group	smit importvg	
Export a volume group	<ol style="list-style-type: none"> 1. Unmount files systems on logical volumes in the volume group: smit umntdsk 2. Vary off the volume group: smit varyoffvg 3. Export the volume group: smit exportvg 	



Attention: A volume group that has a paging space volume on it cannot be exported while the paging space is active. Before exporting a volume group with an active paging space, ensure that

the paging space is not activated automatically at system initialization by typing the following command:

```
chps -a n paging_space name
```

Then, reboot the system so that the paging space is inactive.

Migrating the contents of a physical volume

To move the physical partitions belonging to one or more specified logical volumes from one physical volume to one or more other physical volumes in a volume group, use the following instructions. You can also use this procedure to move data from a failing disk before replacing or repairing the failing disk. This procedure can be used on physical volumes in either the root volume group or a user-defined volume group.



Attention: When the boot logical volume is migrated from a physical volume, the boot record on the source must be cleared or it could cause a system hang. When you execute the **bosboot** command, you must also execute the **chpv -c** command described in step 4 of the following procedure.

1. If you want to migrate the data to a new disk, do the following steps. Otherwise, continue with step 2.
 - a. Check that the disk is recognizable by the system and available by typing:

```
lsdev -Cc disk
```

The output resembles the following:

```
hdisk0 Available 10-60-00-8,0 16 Bit LVD SCSI Disk Drive
hdisk1 Available 10-60-00-9,0 16 Bit LVD SCSI Disk Drive
hdisk2 Available 10-60-00-11,0 16 Bit LVD SCSI Disk Drive
```

- b. If the disk is listed and in the available state, check that it does not belong to another volume group by typing:

```
lspv
```

The output looks similar to the following:

hdisk0	0004234583aa7879	rootvg	active
hdisk1	00042345e05603c1	none	active
hdisk2	00083772caa7896e	imagesvg	active

In the example, **hdisk1** can be used as a destination disk because the third field shows that it is not being used by a volume group.

If the new disk is not listed or unavailable, you need to configure the disk or logical volume storage.

- c. Add the new disk to the volume group by typing:

```
extendvg VGName diskname
```

Where *VGName* is the name of your volume group and *diskname* is the name of the new disk. In the example shown in the previous step, *diskname* would be replaced by **hdisk1**.

2. The source and destination physical volumes must be in the same volume group. To determine whether both physical volumes are in the volume group, type:

```
lsvg -p VGname
```

Where *VGname* is the name of your volume group. The output for a root volume group looks similar to the following:

rootvg:				
PV_NAME	PV STATE	TOTAL PPs	FREE PPs	FREE DISTRIBUTION
hdisk0	active	542	85	00..00..00..26..59
hdisk1	active	542	306	00..00..00..00..06

Note the number of FREE PPs.

3. Check that you have enough room on the target disk for the source that you want to move:
 - a. Determine the number of physical partitions on the source disk by typing:

```
lspv SourceDiskName | grep "USED PPs"
```

Where *SourceDiskName* is of the name of the source disk, for example, hdisk0. The output looks similar to the following:

```
USED PPs:      159 (636 megabytes)
```

In this example, you need 159 FREE PPs on the destination disk to successfully complete the migration.

- b. Compare the number of USED PPs from the source disk with the number of FREE PPs on the destination disk or disks (step 2). If the number of FREE PPs is larger than the number of USED PPs, you have enough space for the migration.
4. Follow this step only if you are migrating data from a disk in the rootvg volume group. If you are migrating data from a disk in a user-defined volume group, proceed to step 5.

Check to see if the boot logical volume (**hd5**) is on the source disk by typing:

```
lspv -l SourceDiskNumber | grep hd5
```

If you get no output, the boot logical volume is not located on the source disk. Continue to step 5.

If you get output similar to the following:

```
hd5          2    2    02..00..00..00..00    /blv
```

then run the following command:

```
migratepv -l hd5 SourceDiskName DestinationDiskName
```

You will receive a message warning you to perform the **bosboot** command on the destination disk. You must also perform a **mkboot -c** command to clear the boot record on the source. Type the following sequence of commands:

```
bosboot -a -d /dev/DestinationDiskName  
bootlist -m normal DestinationDiskName  
mkboot -c -d /dev/SourceDiskName
```

5. Migrate your data by typing the following SMIT fast path:

```
smit migratepv
```

6. List the physical volumes, and select the source physical volume you examined previously.
7. Go to the **DESTINATION** physical volume field. If you accept the default, all the physical volumes in the volume group are available for the transfer. Otherwise, select one or more disks with adequate space for the partitions you are moving (from step 4).
8. If you wish, go to the Move only data belonging to this **LOGICAL VOLUME** field, and list and select a logical volume. You move only the physical partitions allocated to the logical volume specified that are located on the physical volume selected as the source physical volume.
9. Press Enter to move the physical partitions.

At this point, the data now resides on the new (destination) disk. The original (source) disk, however, remains in the volume group. If the disk is still reliable, you could continue to use it as a hot spare disk. Especially when a disk is failing, it is advisable to do the following steps:

1. To remove the source disk from the volume group, type:

```
reducevg VGName SourceDiskName
```

2. To physically remove the source disk from the system, type:

```
rmdev -l SourceDiskName -d
```

Configuring a disk

You can configure a new disk by various methods.

You can configure a new disk in any of the following ways.

- If you can shut down and power off the system, use Method 1. Whenever possible, it is always preferable to shut down and power off any system when you are attaching a physical disk to it.
- If you cannot shut down your system and you know details about the new disk, such as the subclass, type, parent name, and where it is connected, use Method 2.
- If you cannot shut down your system and you only know the location of the disk, use Method 3.

After a disk is configured, although it is generally available for use, the Logical Volume Manager requires that it is further identified as a physical volume.

Method 1

Use the following method when you can shut down and power off the system before attaching the disk:

1. Physically connect the new disk to the system and then power on the disk and system according to the documentation that came with your system.
2. During system boot, let the Configuration Manager (**cfgmgr**) automatically configure the disk.
3. After system boot, with root authority, type the **lspv** command at the command line to look for the new disk's name.

The system returns an entry similar to one of the following:

```
hdisk1  none                none
```

or:

```
hdisk1  00005264d21adb2e    none
```

The first field identifies the system-assigned name of the disk. The second field displays the physical volume ID (PVID), if any. If the new disk does not appear in the **lspv** output, refer to the *Installation and migration*.

At this point, the disk is usable by the system but it needs a PVID for use by the LVM. If the new disk does not have a PVID, then see [“Making an available disk a physical volume”](#) on page 20.

Method 2

Use the following method when you cannot shut down your system and you know the following information about the new disk:

- How the disk is attached (subclass)
- The type of the disk (type)
- Which system attachment the disk is connected to (parent name)
- The logical address of the disk (where connected).

Do the following:

1. Physically connect the new disk to the system and then power on the disk and system according to the documentation that came with your system.
2. To configure the disk and ensure that it is available as a physical volume, use the **mkdev** command with the flags shown, as in the following example:

```
mkdev -c disk -s scsi -t 2200mb -p scsi3 \  
-w 6,0 -a pv=yes
```

This example adds a 2.2 GB disk with a SCSI ID of 6 and logical unit number of 0 to the scsi3 SCSI bus. The **-c** flag defines the class of the device. The **-s** flag defines the subclass. The **-t** flag defines the type of device. The **-p** flag defines the parent device name that you want to assign. The **-w** flag designates the disk's location by SCSI ID and logical unit number. The **-a** flag specifies the device attribute-value pair, **pv=yes**, which makes the disk a physical volume and writes a boot record with a unique physical volume identifier onto the disk (if it does not already have one).

At this point, the disk is defined both as an available device and as a physical volume. You can type the **lspv** command on the command line to list the new disk entry. If the new disk does not appear in the **lspv** output, refer to the *Installation and migration*.

Method 3

Use the following method when you cannot shut down your system and you know only the location of the disk:

1. Physically connect the new disk to the system and then power on the disk and system according to the documentation that came with your system.
2. To check which physical disks are already configured on the system, type the **lspv** command on the command line. For more information about the **lspv** command, see the [lspv command](#) topic. The output looks similar to the following:

```
hdisk0      000005265ac63976    rootvg
```

3. Type **cfgmgr** on the command line to enter the Configuration Manager. The Configuration Manager automatically detects and configures all newly connected devices on the system, including the new disk. For more information about the **cfgmgr** command, see [cfgmgr](#).
4. To confirm that the new disk was configured, type the **lspv** command again. The output looks similar to one of the following:

```
hdisk1      none                none
```

or

```
hdisk1      00005264d21adb2e    none
```

The first field identifies the system-assigned name of the disk. The second field displays the physical volume ID (PVID), if any. If the new disk does not appear in the **lspv** output, refer to the *Installation and migration*.

At this point, the disk is usable by the system but it needs a PVID for use by the LVM. If the new disk does not have a PVID, then see [“Making an available disk a physical volume” on page 20](#).

Making an available disk a physical volume

A disk must be configured as a physical volume before it can be assigned to volume groups and used by the LVM.

Use the following instructions to configure a physical volume:

1. Ensure the disk is known to the operating system, is available, and is not being used by the operating system or any applications. Type the **lspv** command on the command line. The output looks similar to the following:

```
hdisk1  none                none
```

Check the output for the following:

- If the new disk's name does not appear in command output, refer to [“Configuring a disk” on page 19](#).
- If the second field of the output shows a system-generated physical volume identifier (PVID) (for example, 00005264d21adb2e), the disk is already configured as a physical volume and you do not have to complete this procedure.

- If the third field of the output shows a volume group name (for example, `rootvg`), the disk is currently being used and is not an appropriate choice for this procedure.

If the new disk has no PVID and is not in use, continue with the next step.

2. To change an available disk to a physical volume, type the **chdev** command on the command line. For example:

```
chdev -l hdisk3 -a pv=yes
```

The **-l** flag specifies the device name of the disk. The **-a** flag specifies the device attribute-value pair, `pv=yes`, which makes the disk a physical volume and writes a boot record with a unique physical volume identifier onto the disk (if it does not already have one).

At this point, the disk is defined as a physical volume. You can type the **lspv** command on the command line to list the new disk entry.

Changing the PVID and VGID of rootvg

You can change the physical volume identifier (PVID) and the volume group identifier (VGID) of the `rootvg` volume group during the system boot phase.

To change the PVID and VGID of the `rootvg`, set the `sys0 dev ghostdev` attribute with a value of 2 and reboot the system. The `sys0 device ghostdev` attribute is a bitwise flag.

- To set the `sys0 device ghostdev` attribute to change PVID and VGID of `rootvg` volume group, enter the following command:

```
chdev -l sys0 -a ghostdev=2
```

Note: The value of 2 for the `sys0 device ghostdev` attribute is unset, after the **ipl_varyon** command changes the PVID and VGID of all disks in the `rootvg`. If the **chdev** command for changing the PVID of any `rootvg` disks fails, the **ipl_varyon** command sends a warning message and continues to vary on the `rootvg`. If the **chdev** command for changing the PVID of any disk in `rootvg` fails, and you want to change the PVID and VGID during the next reboot, set `sys0 device ghostdev` attribute to 2 again.

- To list the value of the `ghostdev` attribute, enter the following command:

```
lsattr -E -l sys0 -a ghostdev
```

Replacing a failed physical volume in a mirrored volume group

The following procedures replace a failed physical volume (PV) within a mirrored volume group. The **replacepv** command provides a method for replacing a failed PV in most configurations. An alternative procedure is also provided for configurations where the **replacepv** command cannot be used.

Introduction

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

All logical volumes using the failed PV have valid copies on other available PVs, with the possible exception of a dedicated dump logical volume.

Replacing a failed PV using the replacepv command

If any of the following prerequisites cannot be met, see the [alternate procedure](#).

- The volume group containing the failed PV is not `rootvg`.
- The replacement PV can be added to the volume group containing the failed PV (this might not be possible depending on the PV size and volume group characteristics, such as `MAX_PPs_per_PV`).
- The replacement PV must be able to be configured into the system at the same time as the failing PV.

- The replacement PV's name can differ from the failed PV's name.
- The size of the replacement PV must be at least the size of the failed PV.
- The volume group containing the failed PV must not be a snapshot volume group or have a snapshot volume group.

Complete the following steps, assuming that the failed PV is `hdisk2` and the replacement PV is `hdisk10`:

1. If the replacement PV is not yet installed on the system, perform the steps necessary to install it. To use the configuration manager to define a new PV, run the following command:

```
cfgmgr
```

Use the **lspv** command to determine the name assigned to the PV. For this example, assume that the new PV is named `hdisk10`.

2. To replace the failed PV with the one defined in Step 1, run the following command:

```
replacepv hdisk2 hdisk10
```

When the command runs, `hdisk2` is replaced by `hdisk10`, and `hdisk2` is no longer assigned to a volume group.

3. To undefine the failed PV, run the following command:

```
rmdev -dl hdisk2
```

4. Physically remove the failed disk from the system.
5. Verify that the procedure was successful by completing the following steps:
 - a. To check that all logical volumes are mirrored to the new PV as specified, run the following command:

```
lslv lvname
```

Check the **COPIES** attribute of each logical volume affected by the failed PV to ensure that the desired number of copies now exist. If the number of copies of the logical volume is below the desired number, use the **mk1vcopy** command to create additional copies.

- b. To verify that all logical volume partitions are synchronized and there are no stale partitions, run the following command:

```
lspv hdisk10
```

Check the **STALE PARTITIONS** attribute of the replaced PV to ensure that the count is zero. If there are stale partitions use the **syncvg** command to synchronize the partitions.

Step 5 completes the replacement procedure for a failed PV.

Replacing a failed PV when the configuration does not allow the use of the **replacepv** command

Assume that the failed physical volume, `hdisk0`, and its mirror, `hdisk1`, are part of the *yourvg* volume group.

1. To remove mirror copies from the failed PV, run the following command:

```
unmirrorvg yourvg hdisk0
```

2. If the PV failure occurred on `rootvg`, remove `hdisk0` from the boot list by running the following command:

Note: If your configuration uses boot devices other than `hdisk0` and `hdisk1`, add them to the command syntax.

```
bootlist -om normal hdisk1
```

This step requires that `hdisk1` remains a bootable device in `rootvg`. After completing this step, ensure that `hdisk0` does not appear in output.

3. If the PV failure occurred on `rootvg`, recreate any dedicated dump devices from the failed PV.

If you have a dedicated dump device that was on the failed PV, you can use the **mk1v** command to create a new logical volume on an existing PV. Use the **sysdumpdev** command to set the new logical volume as the primary dump device.

4. To undefine the failed PV, run the following command:

Note: Removing the disk device entry will also remove the `/dev/ipldevice` hard link if the failed PV is the PV used to boot the system.

```
reducevg yourvg hdisk0  
rmdev -dl hdisk0
```

5. If the failed PV is the most recently used boot device, recreate the `/dev/ipldevice` hard link that was removed in Step 4 by running the following command:

```
ln /dev/rhdisk1 /dev/ipldevice
```

Note the `r` prefixed to the PV name.

To verify that your `/dev/ipldevice` hard link has been recreated, run the following command:

```
ls /dev/ipldevice
```

6. Replace the failed disk.
7. To define the new PV, run the following command:

```
cfdmgrp
```

The **cfdmgrp** command assigns a PV name to the replacement PV. The assigned PV name is likely to be the same as the PV name previously assigned to the failed PV. In this example, assume that the device `hdisk0` is assigned to the replacement PV.

8. To add the new PV to the volume group, run the following command:

```
extendvg yourvg hdisk0
```

You might encounter the following error message:

```
0516-050 Not enough descriptor space left in this volume group.  
Either try adding a smaller PV or use another volume group.
```

If you encounter this error and cannot add the PV to the volume group, you can try to mirror logical volumes to another PV that already exists in the volume group or add a smaller PV. If neither option is possible, you can try to bypass this limitation by upgrading the volume group to a Big-type or Scalable-type volume group using the **chvg** command.

9. Mirror the volume group.

Note: The **mirrorvg** command cannot be used if all of the following conditions exist:

- The target system is a logical partition (LPAR).
- A copy of the boot logical volume (by default, `hd5`) resides on the failed PV.
- The replacement PV's adapter was dynamically configured into the LPAR since the last cold boot.

If all of the above conditions exist, use the **mk1vcopy** command to recreate mirror copies for each logical volume as follows:

- a. Create copies of the boot logical volume to ensure that it is allocated to a contiguous series of physical partitions.
- b. Create copies of the remaining logical volumes, and synchronize the copies using the **syncvg** command.

- c. Make the disk bootable by shutting down the LPAR and activating it instead of rebooting using the shutdown or reboot commands. This shutdown does not have to be done immediately, but it is necessary for the system to boot from the new PV.

Otherwise, create new copies of logical volumes in the volume group using the new PV with the following command:

Note: The **mirrorvg** command disables quorum by default. For rootvg, you will want to use the **-m** option to ensure that the new logical volume copies are mapped to hdisk0 in the same way as the working disk.

```
mirrorvg yourvg hdisk0
```

10. If your configuration holds copies of some logical volumes, you might need to recreate those copies with the following command:

```
mklvcopy -k
```

11. If the PV failure occurred on rootvg, initialize the boot record by running the following command:

```
bosboot -a
```

12. If the PV failure occurred on rootvg, update the boot list by running the following command:

Note: If your configuration uses boot devices other than hdisk0 and hdisk1, add them to the command.

```
bootlist -om normal hdisk0 hdisk1
```

13. Verify that the procedure was successful.

- To verify that all logical volumes are mirrored to the new PV, run the following command:

```
lslv luname
```

Check the COPIES attribute of each logical volume affected by the failed PV to ensure that the desired number of copies now exist. If the number of copies of the logical volume is below the desired number, use the **mklvcopy** command to create additional copies.

- To verify that all the logical volume partitions are synchronized, check that there are no stale partitions by running the following command:

```
lspv hdisk0
```

Check the STALE PARTITIONS attribute of the replaced PV to ensure that the count is zero. If there are stale partitions use the **syncvg** command to synchronize the partitions.

If the PV failure occurred on rootvg, use the following steps to verify other aspects of this procedure:

- To verify the boot list, run the following command:

```
bootlist -om normal
```

- To verify the dump device, run the following command:

```
sysdumpdev -l
```

- To verify the list of bootable PVs, run the following command:

```
ipl_varyon -i
```

- To verify the /dev/ipl_device, run the following command:

```
ls -i /dev/rhdisk1 /dev/ipldevice
```

Ensure the output of the **ls** command has the same i-node number for both entries.

This step completes the procedure.

Related information

[Logical Volume Manager from A to Z: Introduction and Concepts](#)

Notifying the administrator when a physical volume is missing

Although AIX logs an error when a physical volume becomes inaccessible, there are circumstances in which an error can go undetected.

For example, when the physical volume is part of a mirrored volume group, users do not notice a problem because a good copy of the data remains accessible. In such cases, automatic notification can alert the administrator to the problem before the users notice any disruption to their work.

The following procedure describes how to set up automatic notification when a physical volume is declared missing. By modifying the following procedure, you can track other errors that are significant to you.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. With root authority, make a backup copy of the `/etc/objrepos/errnotify` ODM file.

You can name the backup copy anything that you choose. In the following example, the backup copy appends the `errnotify` file name with the current date:

```
cd /etc/objrepos
cp errnotify errnotifycurrent_date
```

2. Use your favorite editor to create a file named `/tmp/pvmiss.add` that contains the following stanza:

```
errnotify:
    en_pid = 0
    en_name = "LVM_SA_PVMISS"
    en_persistenceflg = 1
    en_label = "LVM_SA_PVMISS"
    en_crcid = 0
    en_type = "UNKN"
    en_alertflg = ""
    en_resource = "LVDD"
    en_rtype = "NONE"
    en_rclass = "NONE"
    en_method = "/usr/lib/ras/pvmiss.notify $1 $2 $3 $4 $5 $6 $7 $8 $9"
```

After you complete all the steps in this topic, the error notification daemon will automatically expand the `$1` through `$9` in this script with detailed information from the error log entry within the notification message.

3. Use your favorite editor to create a file named `/usr/lib/ras/pvmiss.notify` with the following contents:

```
#!/bin/ksh
exec 3>/dev/console
print -u3 "?"
print -u3 - "-----"
print -u3 "ALERT! ALERT! ALERT! ALERT! ALERT! ALERT!"
print -u3 ""
print -u3 "Desc: PHYSICAL VOLUME IS MISSING. SEE ERRPT."
print -u3 ""
print -u3 "Error label: $9"
print -u3 "Sequence number: $1"
print -u3 "Error ID: $2"
print -u3 "Error class: $3"
print -u3 "Error type: $4"
print -u3 "Resource name: $6"
print -u3 "Resource type: $7"
print -u3 "Resource class: $8"
print -u3 - "-----"
print -u3 "?"
mail - "PHSYICAL VOLUME DECLARED MISSING" root <<-EOF
-----
ALERT! ALERT! ALERT! ALERT! ALERT! ALERT!
Desc: PHYSICAL VOLUME IS MISSING. SEE ERRPT.
```

```
Error label: $9
Sequence number: $1
Error ID: $2
Error class: $3
Error type: $4
Resource name: $6
Resource type: $7
Resource class: $8
-----
EOF
```

4. Save your file and exit the editor.
5. Set the appropriate permissions on the file that you created.
For example:

```
chmod 755 /usr/lib/ras/pvmiss.notify
```

6. Type the following command to add the LVM_SA_PVMISS definition that you created in step 2 to the ODM:

```
odmadd /tmp/pvmiss.add
```

Now, the system runs the `/usr/lib/ras/pvmiss.notify` script whenever an LVM_SA_PVMISS error occurs. This script sends a message to the console and also sends mail to the root user.

Related concepts

[Logical volume storage](#)

Logical volumes are groups of information located on physical volumes.

Related information

[odmadd command](#)

Splitting a mirrored disk from a volume group

Snapshot support helps you protect the consistency of your mirrored volume groups from potential disk failure.

Using the snapshot feature, you can split off a mirrored disk or disks to use as a reliable (from the standpoint of the LVM metadata) point-in-time backup of a volume group, and, when needed, reliably reintegrate the split disks into the volume group. In the following procedure, you first split off a mirrored disk from a volume group and then you merge the split-off disk into the original volume group. To further ensure the reliability of your snapshot, file systems must be unmounted and applications that use raw logical volumes must be in a known state (a state from which the application can recover if you need to use the backup).

A volume group cannot be split if any one of the following is true:

- A disk is already missing.
- The last non-stale partition would be on the split-off volume group.
- Any stale partitions exist in the volume group, unless you use the force flag (**-f**) with the **splitvg** command.

Furthermore, the snapshot feature (specifically, the **splitvg** command) cannot be used in enhanced or classic concurrent mode. The split-off volume group cannot be made concurrent or enhanced concurrent and there are limitations to the changes allowed for both the split-off and the original volume group. For details, read the [chvg](#) command description.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. Ensure that the volume group is fully mirrored and that the mirror exists on a disk or set of disks that contains only this set of mirrors.
2. To enable snapshot support, split off the original volume group (**origVG**) to another disk or set of disks, using the following command:

```
splitvg origVG
```

At this point, you now have a reliable point-in-time backup of the original volume group. Be aware, however, that you cannot change the allocation on the split-off volume group.

3. Reactivate the split-off disk and merge it into the original volume group using the following command:

```
joinvg origVG
```

At this point, the split-off volume group is now reintegrated with the original volume group.

Related concepts

[Logical volume storage](#)

Logical volumes are groups of information located on physical volumes.

Related information

[chvg command](#)

[recreatevg command](#)

[splitvg command](#)

[Logical Volume Manager from A to Z: Introduction and Concepts](#)

Troubleshooting LVM

There are several common types of problems with LVM that you can troubleshoot.

Troubleshooting disk drive problems

This information tells how to diagnose and fix disk drive problems.

If you suspect a disk drive is mechanically failing or has failed, run diagnostics on the disk use the following procedure:

1. With root authority, type the following SMIT fast path on the command line:

```
smit diag
```

2. Select **Current Shell Diagnostics** to enter the AIX Diagnostics tool.
3. After you read the Diagnostics Operating Instructions screen, press Enter.
4. Select **Diagnostics Routines**.
5. Select **System Verification**.
6. Scroll down through the list to find and select the drive you want to test.
7. Select **Commit**.

Based on the diagnostics results, you should be able to determine the condition of the disk:

- If you detect the disk drive is failing or has failed, of primary importance is recovering the data from that disk. Migration is the preferred way to recover data from a failing disk. The following procedures describe how to recover or restore data in logical volumes if migration cannot complete successfully.
- If your drive is failing and you can repair the drive without reformatting it, no data will be lost.
- If the disk drive must be reformatted or replaced, make a backup, if possible, and remove the disk drive from its volume group and system configuration before replacing it. Some data from single-copy file systems might be lost.

Disk drive space

If you run out of space on a disk drive, there are several ways to remedy the problem. You can automatically track and remove unwanted files, restrict users from certain directories, or mount space from another disk drive.

You must have root user, system group, or administrative group authority to execute these tasks.

Command for cleaning up file systems automatically

Use the **skulker** command to clean up file systems by removing unwanted files.

Type the following from the command line:

```
skulker -p
```

The **skulker** command is used to periodically purge obsolete or unneeded files from file systems. Candidates include files in the /tmp directory, files older than a specified age, a.out files, core files, or ed.hup files. For more information about the **skulker** command, see **skulker**.

The **skulker** command is typically run daily, as part of an accounting procedure run by the **cron** command during off-peak hours.

Restricting users from certain directories

You can release disk space and possibly keep it free by restricting access to directories and monitoring disk usage.

1. Restrict users from certain directories by typing:

```
chmod 755 DirName
```

This command sets read and write permissions for the owner (root) and sets read-only permissions for the group and others. *DirName* is the full path name of the directory you want to restrict.

2. Monitor the disk usage of individual users by adding the following line to the **/var/spool/cron/crontabs/adm** file:

```
0 2 * * 4 /usr/sbin/acct/dodisk
```

This line runs the **dodisk** command at 2 a.m. (0 2) each Thursday (4). The **dodisk** command initiates disk-usage accounting. This command is usually run as part of an accounting procedure that is run by the **cron** command during off-peak hours.

Mounting space from another disk drive

You can get more space on a disk drive by mounting space from another drive.

You can mount space from one disk drive to another in the following ways:

- Use the **smit mountfs** fast path.
- Use the **mount** command. For example:

```
mount -n nodeA -vnfs /usr/spool /usr/myspool
```

The **mount** command makes a file system available for use at a specific location.

Disk drive recovery without reformatting

If you repair a bad disk and place it back in the system without reformatting it, you can let the system automatically activate and resynchronize the stale physical partitions on the drive at boot time. A *stale physical partition* contains data your system cannot use.

If you suspect a stale physical partition, type the following on the command line:

```
lspv -M PhysVolName
```

Where *PhysVolName* is the name of your physical volume. The **lspv** command output will list all partitions on your physical volume. The following is an excerpt from example output:

hdisk16:112	lv01:4:2	stale
hdisk16:113	lv01:5:2	stale
hdisk16:114	lv01:6:2	stale
hdisk16:115	lv01:7:2	stale
hdisk16:116	lv01:8:2	stale
hdisk16:117	lv01:9:2	stale
hdisk16:118	lv01:10:2	stale

The first column displays the physical partitions and the second column displays the logical partitions. Any stale physical partitions are noted in the third column.

Recovering by using a reformatted or replacement disk drive

You can recover data from a failed disk drive when you must reformat or replace the failed disk.



Attention: Before you reformat or replace a disk drive, remove all references to non-mirrored file systems from the failing disk and remove the disk from the volume group and system configuration. If you do not, you create problems in the ODM (object data manager) and system configuration databases. Instructions for these essential steps are included in the following procedure, under [Before replacing or reformatting your failed or failing disk](#).

The following procedure uses a scenario in which the volume group called *myvg* contains three disk drives that are called *hdisk2*, *hdisk3*, and *hdisk4*. In this scenario, *hdisk3* goes bad. The non-mirrored logical volume *lv01* and a copy of the *mylv* logical volume is contained on *hdisk2*. The *mylv* logical volume is mirrored and has three copies, each of which takes up two physical partitions on its disk. The failing *hdisk3* contains another copy of *mylv*, and the non-mirrored logical volume called *lv00*. Finally, *hdisk4* contains a third copy of *mylv* as well as a logical volume called *lv02*. The following figure shows this scenario.



This procedure is divided into the following key segments:

- The things that you do to protect data before you replace or reformat your failing disk
- The procedure that you follow to reformat or replace the disk
- The things that you do to recover the data after the disk is reformatted or replaced

Before you replace or reformat your failed or failing disk:

1. Log in with root authority.
2. If you are not familiar with the logical volumes that are on the failing drive, use an operational disk to view the contents of the failing disk.

For example, to use *hdisk4* to look at *hdisk3*, type the following on the command line:

```
lspv -M -n hdisk4 hdisk3
```

The **lspv** command displays information about a physical volume within a volume group. The output looks similar to the following:

```
hdisk3:1      mylv:1
hdisk3:2      mylv:2
hdisk3:3      lv00:1
hdisk3:4-50
```

The first column displays the physical partitions, and the second column displays the logical partitions. Partitions 4 through 50 are free.

3. Back up all single-copy logical volumes on the failing device, if possible. For instructions, see [Backing up user files or file systems](#).

4. If you have single-copy file systems, unmount them from the disk.

(You can identify single-copy file systems from the output of the **lspv** command. Single-copy file systems have the same number of logical partitions as physical partitions on the output.) Mirrored file systems do not have to be unmounted.

In the scenario, `lv00` on the failing disk `hdisk3` is a single-copy file system. To unmount it, type the following:

```
umount /dev/lv00
```

If you do not know the name of the file system, assuming the `/etc/filesystems` file is not solely on the failed disk, type `mount` on the command line to list all mounted file systems and find the name that is associated with your logical volume. You can also use the **grep** command on the `/etc/filesystems` file to list only the file system names, if any, associated with your logical volume. For example:

```
grep lv00 /etc/filesystems
```

The output looks similar to the following example:

```
dev          = /dev/lv00
log          = /dev/loglv00
```

Notes:

- a. The **umount** command fails if the file system you are trying to unmount is being used. The **umount** command runs only if none of the file system's files are open and no user's current directory is on that device.
 - b. Another name for the **umount** command is **umount**. The names are interchangeable.
5. Remove all single-copy file systems from the failed physical volume by typing the **rmfs** command:

```
rmfs /FSname
```

6. Remove all mirrored logical volumes on the failing disk.

Note: You cannot use **rm1vcopy** on the `hd5` and `hd7` logical volumes from physical volumes in the `rootvg` volume group. The system does not allow you to remove these logical volumes because there is only one copy of these.

The **rm1vcopy** command removes copies from each logical partition. For example, type:

```
rm1vcopy mylv 2 hdisk3
```

By removing the copy on `hdisk3`, you reduce the number of copies of each logical partition belonging to the `mylv` logical volume from three to two (one on `hdisk4` and one on `hdisk2`).

7. If the failing disk was part of the root volume group and contained logical volume `hd7`, remove the primary dump device (`hd7`) by typing the following on the command line:

```
sysdumpdev -P -p /dev/sysdumpnull
```

The **sysdumpdev** command changes the primary or secondary dump device location for a running system. When you reboot, the dump device returns to its original location.

Note: You can choose to dump to a DVD device. For more information on how to configure a DVD to be the dump device, see **sysdumpdev**.

8. Remove any paging space on the disk by using the following command:

```
rmfs PSname
```

Where *PSname* is the name of the paging space to be removed, which is actually the name of the logical volume on which the paging space resides.

If the **rmpps** command is not successful, you must use the **smit chps** fast path to deactivate the primary paging space and reboot before you continue with this procedure. The **reducevg** command in step 10 can fail if there are active paging spaces.

9. Remove any other logical volumes from the volume group, such as the logical volumes that do not contain a file system, by using the **rm1v** command.
For example, type:

```
rm1v -f lv00
```

10. Remove the failed disk from the volume group by using the **reducevg** command.
For example, type:

```
reducevg -df myvg hdisk3
```

If you cannot run the **reducevg** command or if the command is unsuccessful, the procedure in step 13 can help clean up the VGDA/ODM information after you reformat or replace the drive.

Replacing or reformatting your failed or failing disk:

11. The next step depends on whether you want to reformat or replace your disk and on what type of hardware you are using:

- If you want to reformat the disk drive, use the following procedure:
 - a. With root authority, type the following SMIT fast path on the command line:

```
smit diag
```

- b. Select **Current Shell Diagnostics** to enter the AIX Diagnostics tool.
- c. After you read the **Diagnostics Operating Instructions** screen, press Enter.
- d. Select **Task Selection**.
- e. Scroll down through the task list to find and select **Format Media**.
- f. Select the disk that you want to reformat. After you confirm that you want to reformat the disk, all content on the disk will be erased.

After the disk is reformatted, continue with step 12.

- If your system supports hot swap disks, use the procedure in [“Recovering from disk failure while the system remains available”](#) on page 33, then continue with step 13.
- If your system does not support hot swap disks, do the following steps:
 - Power off the old drive by using the SMIT fast path **smit rmdvsk**. Change the KEEP definition in database field to No.
 - Contact your next level of system support to replace the disk drive.

After you replace or reformat your failed or failing disk:

12. Follow the instructions in [“Configuring a disk”](#) on page 19 and [“Making an available disk a physical volume”](#) on page 20.
13. If you cannot use the **reducevg** command on the disk from the old volume group before the disk was formatted (step 10), the following procedure can help clean up the VGDA/ODM information.
 - a. If the volume group consisted of only one disk that was reformatted, type:

```
exportvg VGName
```

Where *VGName* is the name of your volume group.

- b. If the volume group consists of more than one disk, type the following on the command line:

```
varyonvg VGName
```

The system displays a message about a missing or unavailable disk, and the new (or reformatted) disk is listed. Note the physical volume identifier (PVID) of the new disk, which is listed in the

varyonvg message. It is the 16-character string between the name of the missing disk and the label PVNOTFND. For example:

```
hdisk3 00083772caa7896e PVNOTFND
```

Type:

```
varyonvg -f VGName
```

The missing disk is now displayed with the PVREMOVED label. For example:

```
hdisk3 00083772caa7896e PVREMOVED
```

Then, type the command:

```
reducevg -df VGName PVID
```

Where PVID is the physical volume identifier (in this scenario, 00083772caa7896e).

14. To add the new disk drive to the volume group, use the **extendvg** command.

For example, type:

```
extendvg myvg hdisk3
```

15. To re-create the single-copy logical volumes on the new (or reformatted) disk drive, use the **mk1v** command.

For example, type:

```
mk1v -y lv00 myvg 1 hdisk3
```

This example re-creates the lv00 logical volume on the *hdisk3* drive. The 1 means that this logical volume is not mirrored.

16. To re-create the file systems on the logical volume, use the **crfs** command.

For example, type:

```
crfs -v jfs -d lv00 -m /dev/lv00
```

17. To restore single-copy file system data from backup media, see [Backing up user files or file systems](#).

18. To re-create the mirrored copies of logical volumes, use the **mk1vcopy** command.

For example, type:

```
mk1vcopy mylv 3 hdisk3
```

This example creates a mirrored third partition of the *mylv* logical volume on *hdisk3*.

19. To synchronize the new mirror with the data on the other mirrors (in this example, *hdisk2* and *hdisk4*), use the **syncvg** command.

For example, type:

```
syncvg -p hdisk3
```

As a result, all mirrored file systems must be restored and up-to-date. If you were able to back up your single-copy file systems, they are also ready to use. You must be able to proceed with normal system use.

Example of recovering from a failed disk drive

To recover from a failed disk drive, back out the way you came in; that is, list the steps you went through to create the volume group, and then go backwards.

The following example is an illustration of this technique. It shows how a mirrored logical volume was created and then how it was altered, backing out one step at a time, when a disk failed.

Note: The following example illustrates a specific instance. It is not intended as a general prototype on which to base any general recovery procedures.

1. The system manager, Jane, created a volume group called **workvg** on hdisk1, by typing:

```
mkvg -y workvg hdisk1
```

2. She then created two more disks for this volume group, by typing:

```
extendvg workvg hdisk2  
extendvg workvg hdisk3
```

3. Jane created a logical volume of 40 MB that has three copies.

Each copy is on one of each of the three disks that comprise the **workvg** volume group. She used the following commands:

```
mklv -y testlv workvg 10  
mklvcopy testlv 3
```

After Jane created the mirrored workvg volume group, hdisk2 failed. Therefore, she took the following steps to recover:

1. She removed the logical volume copy from hdisk2 by typing:

```
rmlvcopy testlv 2 hdisk2
```

2. She detached hdisk2 from the system so that the ODM and VGDA are updated, by typing:

```
reducevg workvg hdisk2
```

3. She removed hdisk2 from the system configuration to prepare for replacement by typing:

```
rmdev -l hdisk2 -d
```

4. She chose to shut down the system, by typing:

```
shutdown -F
```

5. She replaced the disk. The new disk did not have the same SCSI ID as the former hdisk2.

6. She rebooted the system.

Because you have a new disk (the system sees that there is a new PVID on this disk), the system chooses the first *open* hdisk name. Because the **-d** flag was used in step 3, the name hdisk2 was released, so the system chose hdisk2 as the name of the new disk. If the **-d** flag had not been used, hdisk4 would have been chosen as the new name.

7. Jane added this disk into the **workvg** volume group by typing:

```
extendvg workvg hdisk2
```

8. She created two mirrored copies of the logical volume by typing:

```
mklvcopy testlv 3
```

The Logical Volume Manager automatically placed the third logical volume copy on the new hdisk2.

Recovering from disk failure while the system remains available

You can recover from disk failure using the hot removability feature.

The procedure to recover from disk failure using the hot removability feature is, for the most part, the same as described in [“Disk drive recovery without reformatting” on page 28](#), with the following exceptions:

1. To unmount file systems on a disk, use the procedure [Mount a JFS or JFS2](#).
2. To remove the disk from its volume group and from the operating system, use the procedure [“Removing a disk without data” on page 50](#).

3. To replace the failed disk with a new one, you do not need to shut down the system. Use the following sequence of procedures:
 - a) [“Logical volume storage” on page 37](#)
 - b) [“Configuring a disk” on page 19](#)
 - c) Continue with step 13 of [“Recovering by using a reformatted or replacement disk drive” on page 29.](#)

Replacing a disk when the volume group consists of one disk

Use one the following procedure if you can access a disk that is going bad as part of a volume group.

- [“Migrating the contents of a physical volume” on page 17](#)

If the disk is bad and cannot be accessed, follow these steps:

1. Export the volume group.
2. Replace the drive.
3. Re-create the data from backup media that exists.

Physical and logical volume errors

There are several common errors with physical and logical volumes that you can troubleshoot.

Hot spot problems

If you notice performance degradation when accessing logical volumes, you might have hot spots in your logical volumes that are experiencing too much disk I/O.

For more information, see [“Hot spot management in logical volumes” on page 63.](#)

LVCB warnings

A warning results if the LVCB contains invalid information.

The logical volume control block (LVCB) is the first block of a logical volume. The size of LVCB is the block size of the physical volumes within the volume group. This area holds important information such as the creation date of the logical volume, information about mirrored copies, and possible mount points in the JFS. Certain LVM commands are required to update the LVCB, as part of the algorithms in LVM. The old LVCB is read and analyzed to see if it is a valid. If the information is valid LVCB information, the LVCB is updated. If the information is not valid, the LVCB update is not performed, and you might receive the following message:

```
Warning, cannot write lv control block data.
```

Most of the time, this message results when database programs bypass the JFS and access raw logical volumes as storage media. When this occurs, the information for the database is literally written over the LVCB. For raw logical volumes, this is not fatal. After the LVCB is overwritten, the user can still:

- Expand a logical volume
- Create mirrored copies of the logical volume
- Remove the logical volume
- Create a journaled file system to mount the logical volume

There are limitations to deleting LVCBs. A logical volume with a deleted LVCB might not import successfully to other systems. During an importation, the LVM **importvg** command scans the LVCBs of all defined logical volumes in a volume group for information concerning the logical volumes. If the LVCB does not exist, the imported volume group still defines the logical volume to the new system that is accessing this volume group, and the user can still access the raw logical volume. However, the following typically happens:

- Any JFS information is lost and the associated mount point is not imported to the new system. In this case, you must create new mount points, and the availability of previous data stored in the file system is not ensured.

- Some non-JFS information concerning the logical volume cannot be found. When this occurs, the system uses default logical volume information to populate the ODM information. Thus, some output from the **lslv** command might be inconsistent with the real logical volume. If any logical volume copies still exist on the original disks, the information will not be correctly reflected in the ODM database. Use the **rm1vcopy** and **mk1vcopy** commands to rebuild any logical volume copies and synchronize the ODM.

Physical partition limits

In the design of Logical Volume Manager (LVM), each logical partition maps to one physical partition (PP). And, each physical partition maps to a number of disk sectors. The design of LVM limits the number of physical partitions that LVM can track per disk to 1016. In most cases, not all of the 1016 tracking partitions are used by a disk.

When this limit is exceeded, you might see a message similar to the following:

```
0516-1162 extndvg: Warning, The Physical Partition Size of PPSize requires the
creation of TotalPPs partitions for PVname. The limitation for volume group
VGname is LIMIT physical partitions per physical volume. Use chvg command
with -t option to attempt to change the maximum Physical Partitions per
Physical volume for this volume group.
```

Where:

PPsize

Is 1 MB to 1 GB in powers of 2.

Total PPs

Is the total number of physical partitions on this disk, given the *PPsize*.

PVname

Is the name of the physical volume, for example, *hdisk3*.

VGname

Is the name of the volume group.

LIMIT

Is 1016 or a multiple of 1016.

This limitation is enforced in the following instances:

1. When creating a volume group using the **mkvg** command, you specified a number of physical partitions on a disk in the volume group that exceeded 1016. To avoid this limitation, you can select from the physical partition size ranges of 1, 2, 4 (the default), 8, 16, 32, 64, 128, 256, 512 or 1024 MB and use the **mkvg -s** command to create the volume group. Alternatively, you can use a suitable factor that allows multiples of 1016 partitions per disk, and use the **mkvg -t** command to create the volume group.
2. When adding a disk to a pre-existing volume group with the **extendvg** command, the new disk caused the 1016 limitation violation. To resolve this situation, convert the existing volume group to hold multiples of 1016 partitions per disk using the **chvg -t** command. Alternatively, you can re-create the volume group with a larger partition size that allows the new disk, or you can create a standalone volume group consisting of a larger physical size for the new disk.

Partition limitations and the rootvg

If the installation code detects that the rootvg drive is larger than 4 GB, it changes the **mkvg -s** value until the entire disk capacity can be mapped to the available 1016 tracks. This installation change also implies that all other disks added to rootvg, regardless of size, are also defined at that physical partition size. The maximum supported size for the rootvg drive is 15.9 TB.

Partition limitations and RAID systems

For systems using a redundant array of identical disks (RAID), the */dev/hdiskX* name used by LVM may consist of many non-4 GB disks. In this case, the 1016 requirement still exists. LVM is unaware of

the size of the individual disks that really make up /dev/hdiskX. LVM bases the 1016 limitation on the recognized size of /dev/hdiskX, and not the real physical disks that make up /dev/hdiskX.

Device configuration database synchronization

A system malfunction can cause the device configuration database to become inconsistent with the LVM. You can synchronize the device configuration database with the LVM information.

When the device configuration database becomes inconsistent with the LVM, a logical volume command generates such error messages as:

```
0516-322 The Device Configuration Database is inconsistent ...
```

OR

```
0516-306 Unable to find logical volume LVname in the Device  
Configuration Database.
```

(where the logical volume called *LVname* is normally available).



Attention: Do not remove the /dev entries for volume groups or logical volumes. Do not change the database entries for volume groups or logical volumes using the Object Data Manager.

To synchronize the device configuration database with the LVM information, with root authority, type the following on the command line:

```
synclvodm -v VGName
```

Where *VGName* is the name of the volume group you want to synchronize.

Fixing volume group errors

Use these methods to fix volume group errors.

If the **importvg** command is not working correctly, try refreshing the device configuration database.

Overriding a vary-on failure



Attention: Overriding a vary-on failure is an unusual operation; check all other possible problem sources such as hardware, cables, adapters, and power sources before proceeding. Overriding a quorum failure during a vary-on process is used only in an emergency and only as a last resort (for example, to salvage data from a failing disk). Data integrity cannot be guaranteed for management data contained in the chosen copies of the VGDA and the VGSA when a quorum failure is overridden.

When you choose to forcibly vary-on a volume group by overriding the absence of a quorum, the PV STATE of all physical volumes that are missing during this vary-on process will be changed to removed. This means that all the VGDA and VGSA copies are removed from these physical volumes. After this is done, these physical volumes will no longer take part in quorum checking, nor are they allowed to become active within the volume group until you return them to the volume group. The varyonvg -f flag (used to override quorum loss) is ignored if the volume group has not lost quorum.

Under one or more of the following conditions, you might want to override the vary-on failure so that the data on the available disks in the volume group can be accessed:

- Unavailable physical volumes appear permanently damaged.
- You can confirm that at least one of the presently accessible physical volumes (which must also contain a good VGDA and VGSA copy) was online when the volume group was last varied on. Unconfigure and power off the missing physical volumes until they can be diagnosed and repaired.

Use the following procedure to avoid losing quorum when one disk is missing or might soon fail and requires repair:

1. To temporarily remove the volume from the volume group, type:

```
chpv -vr PVname
```

When this command completes, the physical volume *PVname* is no longer factored in quorum checking. However, in a two-disk volume group, this command fails if you try the **chpv** command on the disk that contains the two VGDA/VGSAs. The command does not allow you to cause quorum to be lost.

2. If you need to remove the disk for repair, power off the system, and remove the disk. (For instructions, see “[Troubleshooting disk drive problems](#)” on page 27.) After fixing the disk and returning the disk to the system, continue with the next step.
3. To make the disk available again to the volume group for quorum checking, type:

```
chpv -v a PVname
```

Note: The **chpv** command is used only for quorum-checking alteration. The data that resides on the disk is still there and must be moved or copied to other disks if the disk is not to be returned to the system.

VGDA warnings

In some instances, the user experiences a problem adding a new disk to an existing volume group or in creating of a new volume group. The message provided by LVM is:

```
0516-1163 extendvg: VGname already has maximum physical volumes. With the maximum
number of physical partitions per physical volume being LIMIT, the maximum
number of physical volumes for volume group VGname is MaxDisks.
```

Where:

VGname

Is the name of the volume group.

LIMIT

Is 1016 or a multiple of 1016.

MaxDisks

Is the maximum number of disks in a volume group. For example, if there are 1016 physical partitions (PPs) per disk, then *MaxDisk* is 32; if there are 2032, then *MaxDisk* is 16.

You can modify the `image.data` file and then use alternate disk installation, or restore the system using the **mksysb** command to re-create the volume group as a big volume group. For more information, see the *Installation and migration*.

On older AIX versions when the limit was smaller than 32 disks, the exception to this description of the maximum VGDA was the **rootvg**. To provide users with more free disk space, when `rootvg` was created, the **mkvg -d** command used the number of disks selected in the installation menu as the reference number. This **-d** number is 7 for one disk and one more for each additional disk selected. For example, if two disks are selected, the number is 8 and if three disks are selected, the number is 9, and so on.

Logical volume storage

Logical volumes are groups of information located on physical volumes.

A hierarchy of structures is used to manage disk storage. Each individual disk drive, called a *physical volume* (PV) has a name, such as `/dev/hdisk0`. Every physical volume in use belongs to a *volume group* (VG). All of the physical volumes in a volume group are divided into *physical partitions* (PPs) of the same size. For space-allocation purposes, each physical volume is divided into five regions (**outer_edge**, **inner_edge**, **outer_middle**, **inner_middle** and **center**). The number of physical partitions in each region varies, depending on the total capacity of the disk drive.

Within each volume group, one or more *logical volumes* (LVs) are defined. Data on logical volumes appears to be contiguous to the user but can be discontinuous on the physical volume. This allows file systems, paging space, and other logical volumes to be re-sized or relocated, to span multiple physical volumes, and to have their contents replicated for greater flexibility and availability in the storage of data.

Each logical volume consists of one or more *logical partitions* (LPs). Each logical partition corresponds to at least one physical partition. If mirroring is specified for the logical volume, additional physical partitions are allocated to store the additional copies of each logical partition. Although the logical partitions are numbered consecutively, the underlying physical partitions are not necessarily consecutive or contiguous.

Logical volumes can serve a number of system purposes, such as paging, but each logical volume serves a single purpose only. Many logical volumes contain a single journaled file system (JFS or JFS2). Each JFS consists of a pool of pagesize (4 KB) blocks. When data is to be written to a file, one or more additional blocks are allocated to that file. These blocks might not be contiguous with one another or with other blocks previously allocated to the file. A given file system can be defined as having a fragment size of less than 4 KB (512 bytes, 1 KB, 2 KB).

After installation, the system has one volume group (the rootvg volume group) consisting of a base set of logical volumes required to start the system and any other logical volumes you specify to the installation script. Any other physical volumes you have connected to the system can be added to a volume group (using the **extendvg** command). You can add the physical volume either to the rootvg volume group or to another volume group (defined by using the **mkvg** command). Logical volumes can be tailored using the commands, the menu-driven System Management Interface Tool (SMIT) interface.

Related tasks

Notifying the administrator when a physical volume is missing

Although AIX logs an error when a physical volume becomes inaccessible, there are circumstances in which an error can go undetected.

Splitting a mirrored disk from a volume group

Snapshot support helps you protect the consistency of your mirrored volume groups from potential disk failure.

Reducing the size of a file system in your root volume group

The simplest way to reduce *all* file systems to their minimum size is to set the **SHRINK** option to **yes** when restoring the base operating system from backup.

Preparing to install a device

Installing devices on your system consists of identifying where the device is to be attached, connecting the device physically, and configuring the device with the Configuration Manager, or SMIT.

Note: The following procedure requires a shutdown of your system to install the device. Not all device installations require a shutdown of your system. Refer to the documentation shipped with the specific device.

This topic documents installation tasks that are common to all devices. Because of the wide variety of devices that you can install on your system, only a general procedure is provided. For more specific information, see the installation instructions shipped with the specific device.

1. Stop all applications running on the system unit and shut down the system unit using the **shutdown** command.
2. Turn off the system unit and all attached devices.
3. Unplug the system unit and all attached devices.
4. Connect the new device to the system using the procedure described in the setup and operator guide for the device.
5. Plug in the system unit and all attached devices.
6. Turn on all the attached devices leaving the system unit turned off.
7. Turn on the system unit when all the devices complete power-on self-tests (POST).

The Configuration Manager automatically scans the attached devices and configures any new devices it detects. The new devices are configured with default attributes and recorded in the customized configuration database placing the device in **Available** state.

You can manually configure a device using the SMIT fast path, **smit dev**. If you need to customize the device attributes or if the device cannot be configured automatically, see the device documentation that shipped with the device for specific configuration requirements.

Configuring a read/write optical drive

There are two methods for configuring a read/write optical drive.

The read/write optical drive must be connected to the system and powered on.

Method 1

Method one is the faster of the two methods. It only configures the read/write optical drive specified. To use this method, you must provide the following information:

Item	Description
Subclass	Defines how the drive is attached.
Type	Specifies the type of read/write optical drive.
Parent Name	Specifies the system attachment the drive is connected to.
Where Connected	Specifies the logical address of the drive.

Enter the following command to configure the read/write optical drive:

```
mkdev -c rwoptical -s Subclass -t Type -p ParentName -w WhereConnected
```

The following is an example of a read/write optical drive that has a SCSI ID of 6, a logical unit number of zero, and is connected to the third (scsi3) SCSI bus:

```
mkdev -c rwoptical -s scsi -t osomd -p scsi3 -w 6,0 -a pv=yes
```

Method 2

Method two uses the Configuration Manager, searching the current configuration, detecting any new devices, and automatically configuring the devices. This method is used when little information is known about the read/write optical drive.

1. Use the configuration manager to configure all newly detected devices on the system (including the read/write optical drive) by typing:

```
cfgmgr
```

2. Type the following command to list the names, location codes, and types of all currently configured read/write optical drives:

```
lsdev -C -c rwoptical
```

3. Determine the name of the newly configured read/write optical drive using the location code that matches the location of the drive being added.

Configuration of a large number of devices

Devices include hardware components such as, printers, drives, adapters, buses, and enclosures, as well as pseudo-devices, such as the error special file and null special file. Device drivers are located in the `/usr/lib/drivers` directory.

The number of devices that AIX can support can vary from system to system, depending on several important factors. The following factors have an impact on the file systems that support the devices:

- Configuring a large number of devices requires storage of more information in the ODM device-configuration database. It can also require more device special files. As a result, more space and more i-nodes are required of the file system.

- Some devices require more space than others in the ODM device-configuration database. The number of special files or i-nodes used also varies from device to device. As a result, the amount of space and i-nodes required of the file system depends on the types of devices on the system.
- Multipath I/O (MPIO) devices require more space than non-MPIO devices because information is stored in the ODM for the device itself as well as for each path to the device. As a rough guideline, assume that each path takes up the space of one-fifth of a device. For example, an MPIO device with five paths will have the space equivalent to two non-MPIO devices.
- AIX includes both logical devices and physical devices in the ODM device-configuration database. Logical devices include volume groups, logical volumes, network interfaces, and so on. In some cases, the relationship between logical and physical devices can greatly affect the total number of devices supported. For example, if you define a volume group with two logical volumes for each physical disk that is attached to a system, this will result in four AIX devices for each disk. On the other hand, if you define a volume group with six logical volumes for each physical disk, there will be eight AIX devices for each disk. Therefore, only half as many disks could be attached.
- Changing device attributes from their default settings results in a larger ODM device-configuration database and could lead to fewer devices that can be supported.
- More devices require more real memory.

Two file systems are used by AIX to support devices:

- The RAM file system is used during boot in an environment that has no paging space and no disk file systems mounted. The size of the RAM file system is 25% of the system memory size up to a maximum of 128 MB. One i-node is allocated for every KB in the RAM file system. The minimum system memory requirement for the AIX operating system is 256 MB, which translates into a minimum RAM file system size of 64 MB with 65536 i-nodes. If the system memory size is 512 MB or larger, then the RAM file system will be at its maximum size of 128 MB with 131072 i-nodes. If either the amount of RAM file system space or number of i-nodes needed to support the attached devices exceeds what has been allocated to the RAM disk, the system might not boot. If this is the case, you must remove some of the devices.
- The space and i-nodes of the root file system (rootvg) on the disk can be increased as long as there are unallocated partitions in the rootvg. With the maximum RAM file system size, it is likely that up to 25,000 AIX devices can be configured. These numbers include both physical and logical devices. Depending on the various factors mentioned in this section, your system might be able to configure more or fewer devices than this number.

Note: With a large number of devices in the system, the longer configuration time contributes to a longer boot time.

Adding a removable media drive

You can add a removable media drive.

The following procedure uses SMIT to add a CD-ROM drive to your system. Other types of removable media drives are added using different fast paths but all follow the same general procedure. You can also add a removable media drive using the Configuration Manager, or the **mkdev** command.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. To add a CD-ROM drive to your system, install the hardware according to the documentation that came with your system.
2. With root authority, type the following SMIT fast path:

```
smit makcdr
```

3. In the next screen, select the drive type from the available list of supported drives.
4. In the next screen, select the parent adapter from the available list.

5. In the next screen, at minimum, select the connection address from the available list. You can also use this screen to select other options. When you are finished, press Enter, and then SMIT adds the new CD-ROM drive.

At this point, the new CD-ROM drive is recognized by your system. To add a read/write optical drive, use the **smit makomd** fast path. To add a tape drive, use the **smit maktpe** fast path.

Space Reclamation Support for Logical Volume Storage

In AIX 7.2 with the 7200-01 Technology Level, or later, the Logical Volume Manager (LVM) supports space reclamation for physical volumes that are capable of reclaiming space.

LVM informs the disk driver, which in-turn informs the storage subsystem that the partition space is no longer in use and the storage subsystem can reclaim the allocated space. The disk driver helps LVM detect the space reclaim capability of physical volume. LVM and file system configuration commands such as the `rm1v` command, `rm1vcopy` command, and the `chfs(shrink fs)` command initiate the space reclamation for the partitions after they are freed. LVM detects the physical volume's space reclaim capability when it opens the volume during the execution of the `varyonvg` or the `extendvg` command. LVM also tries to detect it while the volume group is online. If the state change detection requires physical volume to be reopened, administrator must run the `varyoffvg` command and then run the `varyonvg` command for the volume group.

Volume groups that are created before AIX 7.2 Technology Level 1 might have free partition space, which is not eligible for automatic reclamation. Administrator can create and delete dummy logical volume on those free partitions to reclaim this space. But space will be automatically reclaimed for the partitions, which are freed after installation of AIX 7.2 Technology Level 1.

The LVM process to reclaim the space runs in the background after a command such as `rm1v` completes execution. If the system crashes before the LVM process completes the reclamation process for all the partitions, then the partitions are freed but the space is not reclaimed for the pending partitions. If this scenario occurs, you can create and delete dummy logical volume to reclaim the space from the remaining partitions.

The LVM process does not delay the processing of the `varyoffvg` command or the `reducevg` command even if the space reclamation process is pending. The space reclamation process is discarded instead of waiting for the process to finish.

Note: Commands wait only for any outstanding space reclaim requests submitted to the disk driver.

The space reclamation functionality is available from the storage subsystem to reclaim the freed space from a physical volume. Each storage subsystem expects the reclaim request to be aligned on specific number of physical blocks, and the number of physical blocks varies according to the storage subsystem. So sometimes reclaiming blocks (all or some) from partition is not possible because reclaim size is not aligned with the physical blocks of the partition. Some storage subsystems support reclaim of block size that is more than the LVM partition size, and partial block reclamation is not possible. In this scenario, LVM might not be able to accumulate enough contiguous free partitions to generate even a single reclaim request. Therefore, when you delete multiple LVM partitions you might not reclaim the equivalent amount of space in the storage subsystem. You can use the `lvmstat` command with the `-x` option to get information about the space reclamation requests generated by the LVM.

Related information

[varyoffvg command](#)

Logical volume storage concepts

The logical volume (which can span physical volumes) is composed of logical partitions allocated onto physical partitions.

The following figure illustrates the relationships among the basic logical storage concepts.

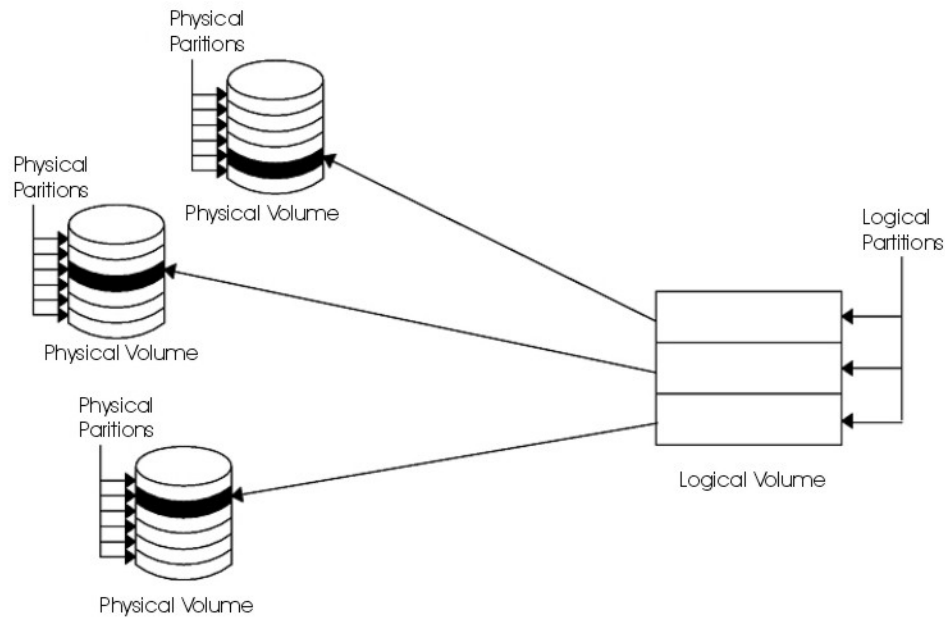


Figure 2. Volume Group

Physical volumes

A disk must be designated as a physical volume and be put into an available state before it can be assigned to a volume group.

A physical volume has certain configuration and identification information written on it. This information includes a physical volume identifier that is unique to the system.

The LVM can make use of the additional space that a redundant array of identical disks (RAID) can add to a logical unit number (LUN), by adding physical partitions to the physical volume associated with the LUN.

Volume groups

A *volume group* is a collection of 1 to 32 physical volumes of varying sizes and types.

A big volume group can have from 1 to 128 physical volumes. A scalable volume group can have up to 1024 physical volumes. A physical volume can belong to only one volume group per system; there can be up to 255 active volume groups.

When a physical volume is assigned to a volume group, the physical blocks of storage media on it are organized into physical partitions of a size you specify when you create the volume group.

When you install the system, one volume group (the root volume group, called **rootvg**) is automatically created that contains the base set of logical volumes required to start the system, as well as any other logical volumes you specify to the installation script. The **rootvg** includes paging space, the journal log, boot data, and dump storage, each in its own separate logical volume. The **rootvg** has attributes that differ from user-defined volume groups. For example, the **rootvg** cannot be imported or exported. When performing a command or procedure on the **rootvg**, you must be familiar with its unique characteristics.

You create a volume group with the **mkvg** command. You add a physical volume to a volume group with the **extendvg** command, make use of the changed size of a physical volume with the **chvg** command, and remove a physical volume from a volume group with the **reducevg** command. Some of the other commands that you use on volume groups include: list (**lsvg**), remove (**exportvg**), install (**importvg**),

reorganize (**reorgvg**), synchronize (**syncvg**), make available for use (**varyonvg**), and make unavailable for use (**varyoffvg**).

Small systems might require only one volume group to contain all the physical volumes attached to the system. You might want to create separate volume groups, however, for security reasons, because each volume group can have its own security permissions. Separate volume groups also make maintenance easier because groups other than the one being serviced can remain active. Because the rootvg must always be online, it contains only the minimum number of physical volumes necessary for system operation.

You can move data from one physical volume to other physical volumes *in the same volume group* with the **migratepv** command. This command allows you to free a physical volume so it can be removed from the volume group. For example, you could move data from a physical volume that is to be replaced.

A volume group that is created with smaller physical and logical volume limits can be converted to a format which can hold more physical volumes and more logical volumes. This operation requires that there be enough free partitions on every physical volume in the volume group for the volume group descriptor area (VGDA) expansion. The number of free partitions required depends on the size of the current VGDA and the physical partition size. Because the VGDA resides on the edge of the disk and it requires contiguous space, the free partitions are required on the edge of the disk. If those partitions are allocated for a user's use, they are migrated to other free partitions on the same disk. The rest of the physical partitions are renumbered to reflect the loss of the partitions for VGDA usage. This renumbering changes the mappings of the logical to physical partitions in all the physical volumes of this volume group. If you have saved the mappings of the logical volumes for a potential recovery operation, generate the maps again after the completion of the conversion operation. Also, if the backup of the volume group is taken with map option and you plan to restore using those maps, the restore operation might fail because the partition number might no longer exist (due to reduction). It is recommended that backup is taken before the conversion and right after the conversion if the map option is used. Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a logical volume, adding a physical volume, and so on) might take considerably longer to run.

Physical partitions

When you add a physical volume to a volume group, the physical volume is partitioned into contiguous, equal-sized units of space called *physical partitions*. A physical partition is the smallest unit of storage space allocation and is a contiguous space on a physical volume.

Physical volumes inherit the volume group's physical partition size, which you can set only when you create the volume group (for example, using the **mkvg -s** command). The following illustration shows the relationship between physical partitions on physical volumes and volume groups.

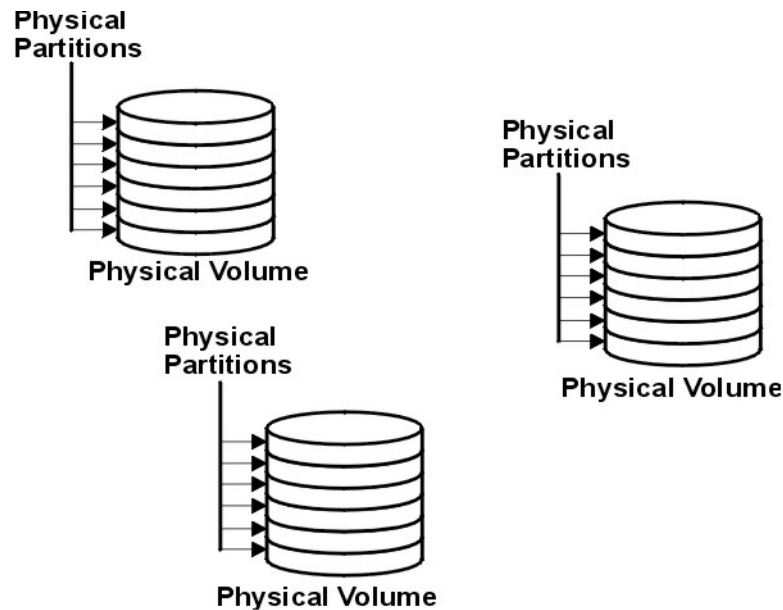


Figure 3. A Volume Group Containing Three Physical Volumes

Logical volumes

After you create a volume group, you can create logical volumes within that volume group.

A *logical volume*, although it can reside on noncontiguous physical partitions or even on more than one physical volume, appears to users and applications as a single, contiguous, extensible disk volume. You can create additional logical volumes with the **mk1v** command. This command allows you to specify the name of the logical volume and define its characteristics, including the number and location of logical partitions to allocate for it.

After you create a logical volume, you can change its name and characteristics with the **ch1v** command, and you can increase the number of logical partitions allocated to it with the **extend1v** command. The default maximum size for a logical volume at creation is 512 logical partitions, unless specified to be larger. The **ch1v** command is used to override this limitation.

Note: After you create a logical volume, the characteristic LV STATE, which can be seen using the **ls1v** command, is closed. It becomes open when, for example, a file system has been created in the logical volume and the logical volume is mounted.

Logical volumes can also be copied with the **cpl1v** command, listed with the **ls1v** command, removed with the **rm1v** command, and have the number of copies they maintain increased or decreased with the **mk1vcopy** and the **rm1vcopy** commands, respectively. Logical Volumes can also be relocated when the volume group is reorganized.

The system allows you to define up to 255 logical volumes per standard volume group (511 for a big volume group and 4095 for a scalable volume group), but the actual number you can define depends on the total amount of physical storage defined for that volume group and the size of the logical volumes you define.

Logical partitions

When you create a logical volume, you specify the number of *logical partitions* for the logical volume.

A logical partition is one, two, or three physical partitions, depending on the number of instances of your data you want maintained. Specifying one instance means there is only one copy of the logical volume (the default). In this case, there is a direct mapping of one logical partition to one physical partition. Each instance, including the first, is termed a *copy*. Where physical partitions are located (that is, how near each other physically) is determined by options you specify when you create the logical volume.

File systems

The logical volume defines allocation of disk space down to the physical-partition level. Finer levels of data management are accomplished by higher-level software components such as the Virtual Memory Manager or the file system. Therefore, the final step in the evolution of a disk is the creation of *file systems*.

You can create one file system per logical volume. To create a file system, use the **crfs** command.

Limitations for logical storage management

The following table shows the limitations for logical storage management.

Although the default maximum number of physical volumes per volume group is 32 (128 for a big volume group, 1024 for a scalable volume group), you can set the maximum for user-defined volume groups when you use the **mkvg** command. For the rootvg, however, this variable is automatically set to the maximum by the system during the installation.

Limitations for Logical Storage Management	
Category	Limit
Volume group	<ul style="list-style-type: none">• 255 volume groups for the 32 bit kernel• 4096 volume groups for the 64 bit kernel <p>Note: The device table on the 64 bit kernel restricts the number of active major numbers to 1024. Consequently, the number of active volume groups is restricted to less than 1024 volume groups.</p>
Physical volume	(MAXPVS/volume group factor) per volume group. MAXPVS is 32 for a standard volume group, 128 for a big volume group, and 1024 for a scalable volume group.
Physical partition	Normal and Big Volume groups: (1016 x volume group factor) per physical volume up to 1024 MB each in size. Scalable volume groups: 2097152 partitions up to 128 GB in size. There is no volume group factor for scalable volume groups.
Logical volume	MAXLVS per volume group, which is 255 for a standard volume group, 511 for a big volume group, and 4095 for a scalable volume group.

If you previously created a volume group before the 1016 physical partitions per physical volume restriction was enforced, stale partitions (no longer containing the most current data) in the volume group are not correctly tracked unless you convert the volume group to a supported state. You can convert the volume group with the **chvg -t** command. A suitable factor value is chosen by default to accommodate the largest disk in the volume group.

For example, if you created a volume group with a 9 GB disk and 4 MB partition size, this volume group will have approximately 2250 partitions. Using a conversion factor of 3 ($1016 * 3 = 3048$) allows all 2250 partitions to be tracked correctly. Converting a standard or big volume group with a higher factor enables inclusion of a larger disk of partitions up to the $1016*$ factor. You can also specify a higher factor when you create the volume group in order to accommodate a larger disk with a small partition size.

These operations reduce the total number of disks that you can add to a volume group. The new maximum number of disks you can add would be a MAXPVS/factor. For example, for a regular volume group, a factor of 2 decreases the maximum number of disks in the volume group to 16 ($32/2$). For a big volume group, a factor of 2 decreases the maximum number of disks in the volume group to 64 ($128/2$).

LVM device size limits

The following limits are the LVM architectural limits. If LVM Bad Block Relocation is required, then PV sizes cannot be larger than 128 GB. For size limitations of specific storage devices, refer to the storage device documentation.

The following size limits are for a 64-bit kernel:

Original VG

PV Limit: 1GB (PP) * 16256 (PPs/PV, factor=16) = 15.9 TB

LV Limit: 1GB (PP) * 32512 (PPs/VG) = 31.8 TB

Big VG

PV Limit: 1GB (PP) * 65024 (PPs/PV, factor=64) = 63.5 TB

LV Limit: 1GB (PP) * 130048 (PPs/VG) = 127 TB

SVG

PV & LV Limit: 128GB (PP) * 2048K (PPs/PV) = 256 PB

The following size limits are for a 32-bit kernel:

All VG Types

PV Limit: < 1 TB

LV Limit: < 1 TB

Configuring Logical Volume Storage

With Logical Volume Storage you can mirror volume groups, define a logical volume, and remove a disk with the system running.

Mirroring a volume group

These scenarios explain how to mirror a normal volume group.

The following instructions show you how to mirror a root volume group using the System Management Interface Tool (SMIT).

(select a volume group in the **Volumes** container, then choose **Mirror** from the **Selected** menu).

Experienced administrators can use the **mirrorvg** command.

1. With root authority, add a disk to the volume group using the following SMIT fast path:

```
smit extendvg
```

2. Mirror the volume group onto the new disk by typing the following SMIT fast path:

```
smit mirrorvg
```

3. In the first panel, select a volume group for mirroring.
4. In the second panel, you can define mirroring options or accept defaults. Online help is available if you need it.

Note: When you complete the SMIT panels and click OK or exit, the underlying command can take a significant amount of time to complete. The length of time is affected by error checking, the size and number of logical volumes in the volume group, and the time it takes to synchronize the newly mirrored logical volumes.

At this point, all changes to the logical volumes will be mirrored as you specified in the SMIT panels.

Mirroring the root volume group

The following scenario explains how to mirror the root volume group (rootvg).

Note: Mirroring the root volume group requires advanced system administration experience. If not done correctly, you can cause your system to be unbootable.

In the following scenario, the rootvg is contained on hdisk01, and the mirror is being made to a disk called hdisk11:

1. Check that hdisk11 is supported by AIX as a boot device:

```
bootinfo -B hdisk11
```

If this command returns a value of 1, the selected disk is bootable by AIX. Any other value indicates that hdisk11 is not a candidate for rootvg mirroring.

2. Extend rootvg to include hdisk11, using the following command:

```
extendvg rootvg hdisk11
```

If you receive the following error messages:

```
0516-050 Not enough descriptor space left in this volume group, Either try
adding a smaller PV or use another volume group.
```

or a message similar to:

```
0516-1162 extendvg: Warning, The Physical Partition size of 16 requires the
creation of 1084 partitions for hdisk11. The limitation for volume group
rootvg is 1016 physical partitions per physical volume. Use chvg command with
the -t option to attempt to change the maximum physical partitions per Physical
Volume for this volume group.
```

You have the following options:

- Mirror the rootvg to an empty disk that already belongs to the rootvg.
- Use a smaller disk.
- Change the maximum number of partitions supported by the rootvg, using the following procedure:
 - a. Check the message for the number of physical partitions needed for the destination disk and the maximum number currently supported by rootvg.
 - b. Use the **chvg -t** command to multiply the maximum number of partitions currently allowed in rootvg (in the above example, 1016) to a number that is larger than the physical partitions needed for the destination disk (in the above example, 1084). For example:

```
chvg -t 2 rootvg
```

- c. Reissue the **extendvg** command at the beginning of step 2.

3. Mirror the rootvg, using the exact mapping option, as shown in the following command:

```
mirrorvg -m rootvg hdisk11
```

This command will turn off quorum when the volume group is rootvg. If you do not use the exact mapping option, you must verify that the new copy of the boot logical volume, hd5, is made of contiguous partitions.

4. Initialize all boot records and devices, using the following command:

```
bosboot -a
```

5. Initialize the boot list with the following command:

```
bootlist -m normal hdisk01 hdisk11
```

Note:

- a. Even though the **bootlist** command identifies hdisk11 as an alternate boot disk, it cannot guarantee the system will use hdisk11 as the boot device if hdisk01 fails. In such case, you might have to boot from the product media, select **maintenance**, and reissue the **bootlist** command without naming the failed disk.

- b. If your hardware model does not support the **bootlist** command, you can still mirror the rootvg, but you must actively select the alternate boot disk when the original disk is unavailable.

Defining a raw logical volume for an application

A *raw logical volume* is an area of physical and logical disk space that is under the direct control of an application, such as a database or a partition, rather than under the direct control of the operating system or a file system.

Bypassing the file system can yield better performance from the controlling application, especially from database applications. The amount of improvement, however, depends on factors such as the size of a database or the application's driver.

Note: You will need to provide the application with the character or block special device file for the new raw logical volume, as appropriate. The application will link to this device file when it attempts opens, reads, writes, and so on.



Attention: Each logical volume has a logical volume control block (LVCB) located in the first block. The size of LVCB is the block size of the physical volumes within the volume group. Data begins in the second block of the physical volume. In a raw logical volume, the LVCB is not protected. If an application overwrites the LVCB, commands that normally update the LVCB will fail and generate a message. Although the logical volume might continue to operate correctly and the overwrite can be an allowable event, overwriting the LVCB is not recommended.

The following instructions use SMIT and the command line interface to define a raw logical volume.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. With root authority, find the free physical partitions on which you can create the raw logical volume by typing the following SMIT fast path:

```
smit lspv
```

2. Select a disk.
3. Accept the default in the second dialog (status) and click **OK**.
4. Multiply the value in the **FREE PPs** field by the value in the **PP SIZE** field to get the total number of megabytes available for a raw logical volume on the selected disk.
If the amount of free space is not adequate, select a different disk until you find one that has enough available free space.
5. Exit SMIT.
6. Use the **mk1v** command to create the raw logical volume.

The following command creates a raw logical volume named `lvdb2003` in the `db2vg` volume group using 38 4-MB physical partitions:

```
mk1v -y lvdb2003 db2vg 38
```

Use the **-y** flag to provide a name for the logical volume instead of using a system-generated name.

At this point, the raw logical volume is created. If you list the contents of your volume group, a raw logical volume is shown with the default type, which is `jfs`. This type entry for a logical volume is simply a label. It does not indicate a file system is mounted for your raw logical volume.

Consult your application's instructions on how to open `/dev/rawLVname` and how to use this raw space.

Related concepts

[Configuring Logical Volume Manager](#)

The Logical Volume Manager (LVM) is installed with the base operating system and needs no further configuration. However, disks must be configured and defined as a physical volume before the LVM can use them.

Related information

[mklv command](#)

[Logical Volume Manager from A to Z: Introduction and Concepts](#)

Unmirroring the root volume group

You can unmirror the root volume group.



Attention: Unmirroring the root volume group requires advanced system administration experience. If not done correctly, your system can become unbootable.

In the following scenario, the root volume group is on `hdisk01` and mirrored onto `hdisk11`. This example removes the mirror on `hdisk11`. The procedure is the same, regardless of which disk you booted to last.

1. Use the following command to unmirror the root volume group on `hdisk11`:

```
unmirrorvg rootvg hdisk11
```

The **`unmirrorvg`** command turns quorum back on for the root volume group.

2. Use the following command to reduce the disk out of the root volume group:

```
reducevg rootvg hdisk11
```

3. Use the following command to reinitialize the boot record of the remaining disk:

```
bosboot -a -d /dev/hdisk01
```

4. Use the following command to modify the boot list in order to remove the unmirrored disk from the list:

```
bootlist -m normal hdisk01
```

The disk is unmirrored.

Removing a disk while the system remains available

The following procedure describes how to remove a disk using the hot-removability feature, which lets you remove the disk without turning the system off. This feature is only available on certain systems.

Hot removability is useful when you want to:

- Remove a disk that contains data in a separate non-rootvg volume group for security or maintenance purposes.
- Permanently remove a disk from a volume group.
- Correct a disk failure.

Removing a disk with data

Use this procedure to remove a disk that contains data without turning the system off.

The disk you are removing must be in a separate non-rootvg volume group. Use this procedure when you want to move a disk to another system.

1. To list the volume group associated with the disk you want to remove, type:

```
smit lspv
```

Your output looks similar to the following:

```
PHYSICAL VOLUME:      hdisk2              VOLUME GROUP:      imagesvg
PV IDENTIFIER:        00083772caa7896e  VG IDENTIFIER
0004234500004c00000000e9b5cac262
```

```

PV STATE:          active
STALE PARTITIONS:  0
PP SIZE:           16 megabyte(s)
TOTAL PPs:         542 (8672 megabytes)
FREE PPs:          19 (304 megabytes)
USED PPs:          523 (8368 megabytes)
FREE DISTRIBUTION: 00..00..00..00..19
USED DISTRIBUTION: 109..108..108..108..90
ALLOCATABLE:       yes
LOGICAL VOLUMES:   5
VG DESCRIPTORS:    2
HOT SPARE:         no

```

The name of the volume group is listed in the VOLUME GROUP field. In this example, the volume group is imagesvg.

2. To verify that the disk is in a separate non-rootvg volume group, type:

```
smit lsvg
```

Then select the volume group associated with your disk (in this example, imagesvg). Your output looks similar to the following:

```

VOLUME GROUP:  imagesvg          VG IDENTIFIER:
0004234500004c000000000e9b5cac262

VG STATE:      active
VG PERMISSION: read/write
MAX LVs:       256
LVs:          5
OPEN LVs:      4
TOTAL PVs:     1
STALE PVs:     0
ACTIVE PVs:    1
MAX PPs per PV: 1016
LTG size:      128 kilobyte(s)
HOT SPARE:     no

PP SIZE:       16 megabyte(s)
TOTAL PPs:     542 (8672 megabytes)
FREE PPs:      19 (304 megabytes)
USED PPs:      523 (8368 megabytes)
QUORUM:        2
VG DESCRIPTORS: 2
STALE PPs:     0
AUTO ON:       yes
MAX PVs:       32
AUTO SYNC:     no

```

In this example, the TOTAL PVs field indicates there is only one physical volume associated with imagesvg. Because all data in this volume group is contained on hdisk2, hdisk2 can be removed using this procedure.

3. To unmount any file systems on the logical volumes on the disk, type:

```
smit umountfs
```

4. To deactivate the volume group, type:

```
smit varyoffvg
```

5. To export the volume group, type:

```
smit exportvg
```

6. To remove the disk, type:

```
smit rmvdsk
```

7. Look at the LED display for the disk you want to remove. Ensure the yellow LED is off (not lit).
8. Physically remove the disk. For more information about the removal procedure, see the service guide for your machine.

At this point, the disk is physically and logically removed from your system. If you are permanently removing this disk, this procedure is completed.

Removing a disk without data

The following procedure describes how to remove a disk that contains either no data or no data that you want to keep.



Attention: The following procedure erases any data that resides on the disk.

1. To unmount any file systems on the logical volumes on the disk, type:

```
smit umountfs
```

2. To deactivate the volume group, type:

```
smit varyoffvg
```

3. To export the volume group, type:

```
smit exportvg
```

4. To remove the disk, type:

```
smit rmvdsk
```

5. Look at the LED display for the disk you want to remove. Ensure the yellow LED is off (not lit).
6. Physically remove the disk.

For more information about the removal procedure, see the service guide for your machine.

At this point, the disk is physically and logically removed from your system. If you are permanently removing this disk, this procedure is completed.

Removing a logical volume by removing the file system

The following procedure explains how to remove a JFS or JFS2 file system, its associated logical volume, its associated stanza in the `/etc/filesystems` file, and, optionally, the mount point (directory) where the file system is mounted.



Attention: When you remove a file system, you destroy all data in the specified file systems and logical volume.

If you want to remove a logical volume with a different type of file system mounted on it or a logical volume that does not contain a file system you can remove the logical volume only.

To remove a journaled file system through SMIT, use the following procedure:

1. Unmount the file system that resides on the logical volume with a command similar to the following example:

```
umount /adam/usr/local
```

Note: You cannot use the **umount** command on a device in use. A device is in use if any file is open for any reason or if a user's current directory is on that device.

2. To remove the file system, type the following fast path:

```
smit rmfs
```

- 3.

1. Select the name of the file system you want to remove.
2. Go to the **Remove Mount Point** field and toggle to your preference. If you select **yes**, the underlying command will also remove the mount point (directory) where the file system is mounted (if the directory is empty).
3. Press Enter to remove the file system. SMIT prompts you to confirm whether you want to remove the file system.
4. Confirm you want to remove the file system. SMIT displays a message when the file system has been removed successfully.

At this point, the file system, its data, and its associated logical volume are completely removed from your system.

Related tasks

[Removing a logical volume only](#)

Use this procedure to remove a logical volume with a different type of file system mounted on it or a logical volume that does not contain a file system.

Removing a logical volume only

Use this procedure to remove a logical volume with a different type of file system mounted on it or a logical volume that does not contain a file system.



Attention: Removing a logical volume destroys all data in the specified file systems and logical volume.

The following procedures explain how to remove a logical volume and any associated file system. You can use this procedure to remove a non-JFS file system or a logical volume that does not contain a file system. After the following procedures describe how to remove a logical volume, they describe how to remove any non-JFS file system's stanza in the `/etc/filesystems` file.

To remove a logical volume through SMIT, use the following procedure:

1. If the logical volume does not contain a file system, skip to step 4.
2. Unmount all file systems associated with the logical volume by typing:

```
umount /FSname
```

Where `/FSname` is the full path name of a file system.

Note:

- a. The **umount** command fails if the file system you are trying to **umount** is currently being used. The **umount** command executes only if none of the file system's files are open and no user's current directory is on that device.
 - b. Another name for the **umount** command is **umount**. The names are interchangeable.
3. To list information you need to know about your file systems, type the following fast path:

```
smit lsfs
```

The following is a partial listing:

Name	Nodename	Mount Pt	...
/dev/hd3	--	/tmp	...
/dev/locallv	--	/adam/usr/local	...

4. Assuming standard naming conventions for the second listed item, the file system is named `/adam/usr/local` and the logical volume is `locallv`. To verify this, type the following fast path:

```
smit lslv2
```

The following is a partial listing:

imagesvg:	LV NAME	TYPE	LPs	PPs	PVs	LV STATE	MOUNT POINT
hd3		jfs	4	4	1	open/syncd	/tmp
locallv		mine	4	4	1	closed/syncd	/adam/usr/local

5. To remove the logical volume, type the following fast path on the command line:

```
smit rmlv
```

6. Select the name of the logical volume you want to remove.
7. Go to the **Remove Mount Point** field and toggle to your preference. If you select **yes**, the underlying command will also remove the mount point (directory) where the file system is mounted (if any, and if that directory is empty).
8. Press Enter to remove the logical volume. SMIT prompts you to confirm whether you want to remove the logical volume.

9. Confirm you want to remove the logical volume. SMIT displays a message when the logical volume has been removed successfully.
10. If the logical volume had a non-JFS file system mounted on it, remove the file system and its associated stanza in the `/etc/filesystems` file, as shown in the following example:

```
rmfs /adam/usr/local
```

Or, you can use the file system name as follows:

```
rmfs /dev/locallv
```

At this point, the logical volume is removed. If the logical volume contained a non-JFS file system, that system's stanza has also been removed from the `/etc/filesystems` file.

Related tasks

[Removing a logical volume by removing the file system](#)

The following procedure explains how to remove a JFS or JFS2 file system, its associated logical volume, its associated stanza in the `/etc/filesystems` file, and, optionally, the mount point (directory) where the file system is mounted.

Resizing a RAID volume group

On systems that use a redundant array of independent disks (RAID), **chvg** and **chpv** command options provide the ability to add a disk to the RAID group and grow the size of the physical volume that LVM uses without interruptions to the use or availability of the system.

Note:

1. This feature is not available while the volume group is activated in classic or in enhanced concurrent mode.
2. The rootvg volume group cannot be resized using the following procedure.
3. A volume group with an active paging space cannot be resized using the following procedure.

The size of all disks in a volume group is automatically examined when the volume group is activated (varyon). If growth is detected, the system generates an informational message.

The following procedure describes how to grow disks in a RAID environment:

1. To check for disk growth and resize if needed, type the following command:

```
chvg -g VGname
```

Where *VGname* is the name of your volume group. This command examines all disks in the volume group. If any have grown in size, it attempts to add physical partitions to the physical volume. If necessary, it will determine the appropriate 1016 limit multiplier and convert the volume group to a big volume group.

2. To turn off LVM bad block relocation for the volume group, type the following command:

```
chvg -b ny VGname
```

Where *VGname* is the name of your volume group.

> | Encrypting logical volumes

Starting with IBM AIX 7.2 with Technology Level 5, the Logical Volume Manager (LVM) supports the data encryption at the logical volume (LV) level. For more information about logical volume encryption, see [Encrypted logical volumes](#).

Volume group strategy

Disk failure is the most common hardware failure in the storage system, followed by failure of adapters and power supplies. Protection against disk failure primarily involves the configuration of the logical volumes.

To protect against adapter and power supply failure, consider a special hardware configuration for any specific volume group. Such a configuration includes two adapters and at least one disk per adapter, with mirroring across adapters, and a nonquorum volume group configuration. The additional expense of this configuration is not appropriate for all sites or systems. It is recommended only where high (up-to-the-last second) availability is a priority. Depending on the configuration, high availability can cover hardware failures that occur between the most recent backup and the current data entry. High availability does not apply to files deleted by accident.

When to create separate volume groups

There are several reasons why you might want to organize physical volumes into volume groups separate from rootvg.

- For safer and easier maintenance.
 - Operating system updates, reinstallations, and crash recoveries are safer because you can separate user file systems from the operating system so that user files are not jeopardized during these operations.
 - Maintenance is easier because you can update or reinstall the operating system without having to restore user data. For example, before updating, you can remove a user-defined volume group from the system by unmounting its file systems. Deactivate it using the **varyoffvg** command, then export the group using the **exportvg** command. After updating the system software, you can reintroduce the user-defined volume group using the **importvg** command, then remount its file systems.
- For different physical-partition sizes. All physical volumes within the same volume group must have the same physical partition size. To have physical volumes with different physical partition sizes, place each size in a separate volume group.
- When different quorum characteristics are required. If you have a file system for which you want to create a nonquorum volume group, maintain a separate volume group for that data; all of the other file systems should remain in volume groups operating under a quorum.
- For security. For example, you might want to remove a volume group at night.
- To switch physical volumes between systems. If you create a separate volume group for each system on an adapter that is accessible from more than one system, you can switch the physical volumes between the systems that are accessible on that adapter without interrupting the normal operation of either (see the **varyoffvg**, **exportvg**, **importvg**, and **varyonvg** commands).

High availability in case of disk failure

The primary methods used to protect against disk failure involve logical volume configuration settings, such as mirroring.

While the volume group considerations are secondary, they have significant economic implications because they involve the number of physical volumes per volume group:

- The quorum configuration, which is the default, keeps the volume group active (varied on) as long as a quorum (51%) of the disks is present. In most cases, you need at least three disks with mirrored copies in the volume group to protect against disk failure.
- The nonquorum configuration keeps the volume group active (varied on) as long as one VGDA is available on a disk. With this configuration, you need only two disks with mirrored copies in the volume group to protect against disk failure.

When deciding on the number of disks in each volume group, you must also plan for room to mirror the data. Keep in mind that you can only mirror and move data between disks that are in the same volume group. If the site uses large file systems, finding disk space on which to mirror could become a problem

at a later time. Be aware of the implications on availability of inter-disk settings for logical volume copies and intra-disk allocation for a logical volume.

High availability in case of adapter or power supply failure

To protect against adapter or power supply failure, depending on your requirements, do one or more of the following.

- Use two adapters, located in the same or different chassis. Locating the adapters in different chassis protects against losing both adapters if there is a power supply failure in one chassis.
- Use two adapters, attaching at least one disk to each adapter. This protects against a failure at either adapter (or power supply if adapters are in separate cabinets) by still maintaining a quorum in the volume group, assuming *cross-mirroring* (copies for a logical partition cannot share the same physical volume) between the logical volumes on disk A (adapter A) and the logical volumes on disk B (adapter B). This means that you copy the logical volumes that reside on the disks attached to adapter A to the disks that reside on adapter B and also that you copy the logical volumes that reside on the disks attached to adapter B to the disks that reside on adapter A as well.
- Configure all disks from both adapters into the same volume group. This ensures that at least one logical volume copy remains intact in case an adapter fails, or, if cabinets are separate, in case a power supply fails.
- Make the volume group a nonquorum volume group. This allows the volume group to remain active as long as one Volume Group Descriptor Area (VGDA) is accessible on any disk in the volume group.
- If there are two disks in the volume group, implement cross-mirroring between the adapters. If more than one disk is available on each adapter, implement double-mirroring. In that case, you create a mirrored copy on a disk that uses the same adapter and one on a disk using a different adapter.

Related concepts

Conversion of a volume group to nonquorum status

You can change a volume group to nonquorum status to have data continuously available even when there is no quorum.

Logical volume strategy

The policies described here help you set a strategy for logical volume use that is oriented toward a combination of availability, performance, and cost that is appropriate for your site.

Availability is the ability to access data even if its associated disk has failed or is inaccessible. The data might remain accessible through copies of the data that are made and maintained on separate disks and adapters during normal system operation. Techniques such as mirroring and the use of hot spare disks can help ensure data availability.

Performance is the average speed at which data is accessed. Policies such as write-verify and mirroring enhance availability but add to the system processing load, and thus degrade performance. Mirroring doubles or triples the size of the logical volume. In general, increasing availability degrades performance. Disk striping can increase performance. Disk striping is allowed with mirroring. You can detect and remedy hot-spot problems that occur when some of the logical partitions on your disk have so much disk I/O that your system performance noticeably suffers.

By controlling the allocation of data on the disk and between disks, you can tune the storage system for the highest possible performance. See *Performance management* for detailed information on how to maximize storage-system performance.

Use the following sections to evaluate the trade-offs among performance, availability, and cost. Remember that increased availability often decreases performance, and vice versa. Mirroring may increase performance, however, the LVM chooses the copy on the least busy disk for Reads.

Note: Mirroring does not protect against the loss of individual files that are accidentally deleted or lost because of software problems. These files can only be restored from conventional tape or disk backups.

Requirements for mirroring or striping

Determine whether the data that is stored in the logical volume is valuable enough to warrant the processing and disk-space costs of mirroring. If you have a large sequential-access file system that is performance-sensitive, you may want to consider disk striping.

Performance and mirroring are not always opposed. If the different instances (copies) of the logical partitions are on different physical volumes, preferably attached to different adapters, the LVM can improve Read performance by reading the copy on the least busy disk. Write performance, unless disks are attached to different adapters, always cost the same because you must update all copies. It is only necessary to read one copy for a Read operation.

AIX LVM supports the following RAID options:

Table 3. Logical Volume Manager support for RAID	
Item	Description
RAID 0	Striping
RAID 1	Mirroring
RAID 10 or 0+1	Mirroring and striping

While mirroring improves storage system availability, it is not intended as a substitute for conventional tape backup arrangements.

You can mirror the rootvg, but if you do, create a separate dump logical volume. Dumping to a mirrored logical volume can result in an inconsistent dump. Also, because the default dump device is the primary paging logical volume, create a separate dump logical volume if you mirror your paging logical volumes.

Normally, whenever data on a logical partition is updated, all the physical partitions containing that logical partition are automatically updated. However, physical partitions can become *stale* (no longer containing the most current data) because of system malfunctions or because the physical volume was unavailable at the time of an update. The LVM can refresh stale partitions to a consistent state by copying the current data from an up-to-date physical partition to the stale partition. This process is called *mirror synchronization*. The refresh can take place when the system is restarted, when the physical volume comes back online, or when you issue the **syncvg** command.

Any change that affects the physical partition makeup of a boot logical volume requires that you run the **bosboot** command after that change. This means that actions such as changing the mirroring of a boot logical volume require a **bosboot**.

Scheduling policies for mirrored writes to disk

For data that has only one physical copy, the logical volume device driver (LVDD) translates a logical Read or Write request address into a physical address and calls the appropriate physical device driver to service the request. This single-copy or nonmirrored policy handles bad block relocation for Write requests and returns all Read errors to the calling process.

If you use mirrored logical volumes, the following scheduling policies for writing to disk can be set for a logical volume with multiple copies:

Sequential-scheduling policy

Performs Writes to multiple copies or mirrors in order. The multiple physical partitions representing the mirrored copies of a single logical partition are designated primary, secondary, and tertiary. In sequential scheduling, the physical partitions are written to in sequence. The system waits for the Write operation for one physical partition to complete before starting the Write operation for the next one. When all write operations have been completed for all mirrors, the Write operation is complete.

Parallel-scheduling policy

Simultaneously starts the Write operation for all the physical partitions in a logical partition. When the Write operation to the physical partition that takes the longest to complete finishes, the Write operation is completed. Specifying mirrored logical volumes with a parallel-scheduling policy might

improve I/O read-operation performance, because multiple copies allow the system to direct the read operation to the least busy disk for this logical volume.

Parallel write with sequential read-scheduling policy

Simultaneously starts the Write operation for all the physical partitions in a logical partition. The primary copy of the read is always read first. If that Read operation is unsuccessful, the next copy is read. During the Read retry operation on the next copy, the failed primary copy is corrected by the LVM with a hardware relocation. This patches the bad block for future access.

Parallel write with round robin read-scheduling policy

Simultaneously starts the Write operation for all the physical partitions in a logical partition. Reads are switched back and forth between the mirrored copies.

Bad block policy

Indicates whether the volume group is enabled for bad block relocation. The default value is *yes*. When the value is set to *yes* for the volume group, the bad blocks can be relocated. When the value is set to *no*, the policy overrides the logical volume settings. When the value is changed, all logical volumes continue with the previous setting. The value indicates whether or not a requested I/O must be directed to a relocated block. If the value is set to *yes*, the volume group allows bad block relocation. If the value is set to *no*, bad block allocation is not completed. The LVM performs software relocation only when hardware relocation fails. Otherwise, the LVM bad block relocation (BBR) flag has no effect.

Note: Bad block relocation is disabled unless the bad block policy settings for volume group and logical volume are both set to *yes*.

Mirror Write Consistency policy for a logical volume

When Mirror Write Consistency (MWC) is turned ON, logical partitions that might be inconsistent if the system or the volume group is not shut down properly are identified. When the volume group is varied back online, this information is used to make logical partitions consistent. This is referred to as *active MWC*.

When a logical volume is using active MWC, then requests for this logical volume are held within the scheduling layer until the MWC cache blocks can be updated on the target physical volumes. When the MWC cache blocks have been updated, the request proceeds with the physical data Write operations. Only the disks where the data actually resides must have these MWC cache blocks written to it before the Write can proceed.

When active MWC is being used, system performance can be adversely affected. Adverse effects are caused by the overhead of logging or journaling a Write request in which a Logical Track Group (LTG) is active. The allowed LTG sizes for a volume group are 128 K, 256 K, 512 K, 1024 K, 2 MB, 4 MB, 8 MB, and 16 MB.

Note: To have an LTG size greater than 128 K, the disks contained in the volume group must support I/O requests of this size from the disk's strategy routines. The LTG is a contiguous block contained within the logical volume and is aligned on the size of the LTG. This overhead is for mirrored Writes only.

It is necessary to guarantee data consistency between mirrors only if the system or volume group crashes before the Write to all mirrors has been completed. All logical volumes in a volume group share the MWC log. The MWC log is maintained on the outer edge of each disk. Locate the logical volumes that use Active MWC at the outer edge of the disk so that the logical volume is in close proximity to the MWC log on disk.

When MWC is set to passive, the volume group logs that the logical volume has been opened. After a crash when the volume group is varied on, an automatic force sync of the logical volume is started. Consistency is maintained while the force sync is in progress by using a copy of the read recovery policy that propagates the blocks being read to the other mirrors in the logical volume. This policy is only supported on the BIG volume group type.

When MWC is turned OFF, the mirrors of a mirrored logical volume can be left in an inconsistent state in the event of a system or volume group crash. There is no automatic protection of mirror consistency. Writes outstanding at the time of the crash can leave mirrors with inconsistent data the next time the volume group is varied on. After a crash, any mirrored logical volume that has MWC turned OFF should perform a forced sync before the data within the logical volume is used. For example,

```
syncvg -f -l LTVname
```

An exception to forced sync is logical volumes whose content is only valid while the logical volume is open, such as paging spaces.

A mirrored logical volume is the same as a nonmirrored logical volume with respect to a Write. When LVM completely finishes with a Write request, the data has been written to all the drive(s) below LVM. The outcome of the Write is unknown until LVM issues an `iodone` on the Write. After this is complete, no recovery after a crash is necessary. Any blocks being written that have not been completed (`iodone`) when a machine crashes should be checked and rewritten, regardless of the MWC setting or whether they are mirrored.

Because a mirrored logical volume is the same as a nonmirrored logical volume, there is no such thing as latest data. All applications that care about data validity need to determine the validity of the data of outstanding or in-flight writes that did not complete before the volume group or system crashed, whether or not the logical volume was mirrored.

Active and passive MWC make mirrors consistent only when the volume group is varied back online after a crash by picking one mirror and propagating that data to the other mirrors. These MWC policies do not keep track of the latest data. Active MWC only keeps track of LTGs currently being written, therefore MWC does not guarantee that the latest data will be propagated to all the mirrors. Passive MWC makes mirrors consistent by going into a propagate-on-read mode after a crash. It is the application above LVM that has to determine the validity of the data after a crash. From the LVM perspective, if the application always reissues all outstanding Writes from the time of the crash, the possibly inconsistent mirrors will be consistent when these Writes finish, (as long as the same blocks are written after the crash as were outstanding at the time of the crash).

Note: Mirrored logical volumes containing either JFS logs or file systems must be synchronized after a crash either by forced sync before use, by turning MWC on, or turning passive MWC on.

Inter-disk allocation policies

The inter-disk allocation policy specifies the number of disks on which the physical partitions of a logical volume are located.

The physical partitions for a logical volume might be located on a single disk or spread across all the disks in a volume group. The following options are used with the **mk1v** and **ch1v** commands to determine inter-disk policy:

- The `Range` option determines the number of disks used for a single physical copy of the logical volume.
- The `Strict` option determines whether the **mk1v** operation succeeds if two or more copies must occupy the same physical volume.
- The `Super Strict` option specifies that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror.
- Striped logical volumes can only have a maximum range and a super strict inter-disk policy.

Inter-disk settings for a single copy of the logical volume

If you select the minimum inter-disk setting (`Range = minimum`), the physical partitions assigned to the logical volume are located on a single disk to enhance availability. If you select the maximum inter-disk setting (`Range = maximum`), the physical partitions are located on multiple disks to enhance performance.

For nonmirrored logical volumes, use the minimum setting to provide the greatest availability (access to data in case of hardware failure). The minimum setting indicates that one physical volume contains all the original physical partitions of this logical volume if possible. If the allocation program must use two or more physical volumes, it uses the minimum number, while remaining consistent with other parameters.

By using the minimum number of physical volumes, you reduce the risk of losing data because of a disk failure. Each additional physical volume used for a single physical copy increases that risk. A nonmirrored logical volume spread across four physical volumes is four times as likely to lose data because of one physical volume failure than a logical volume contained on one physical volume.

The following figure illustrates a minimum inter-disk allocation policy.

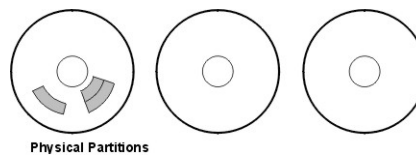


Figure 4. Minimum Inter-Disk Allocation Policy

This illustration shows three disks. One disk contains three physical partitions; the others have no physical partitions.

The maximum setting, considering other constraints, spreads the physical partitions of the logical volume as evenly as possible over as many physical volumes as possible. This is a performance-oriented option, because spreading the physical partitions over several disks tends to decrease the average access time for the logical volume. To improve availability, the maximum setting is only used with mirrored logical volumes.

The following figure illustrates a maximum inter-disk allocation policy.

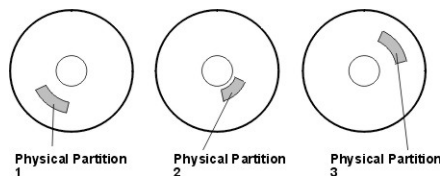


Figure 5. Maximum Inter-Disk Allocation Policy

This illustration shows three disks, each containing one physical partition.

These definitions are also applicable when extending or copying an existing logical volume. The allocation of new physical partitions is determined by your current allocation policy and where the existing used physical partitions are located.

Related concepts

Inter-disk settings for logical volume copies

The allocation of a single copy of a logical volume on disk is fairly straightforward.

Inter-disk settings for logical volume copies

The allocation of a single copy of a logical volume on disk is fairly straightforward.

When you create mirrored copies, however, the resulting allocation is somewhat complex. The figures that follow show minimum maximum and inter-disk (Range) settings for the first instance of a logical volume, along with the available Strict settings for the mirrored logical volume copies.

For example, if there are mirrored copies of the logical volume, the minimum setting causes the physical partitions containing the first instance of the logical volume to be allocated on a single physical volume, if possible. Then, depending on the setting of the Strict option, the additional copy or copies are allocated on the same or on separate physical volumes. In other words, the algorithm uses the minimum number of physical volumes possible, within the constraints imposed by other parameters such as the Strict option, to hold all the physical partitions.

The setting `Strict = y` means that each copy of the logical partition is placed on a different physical volume. The setting `Strict = n` means that the copies are not restricted to different physical volumes. By comparison, the `Super Strict` option would not allow any physical partition from one mirror to be on the same disk as a physical partition from another mirror of the same logical volume.

Note: If there are fewer physical volumes in the volume group than the number of copies per logical partition you have chosen, set Strict to **n**. If Strict is set to **y**, an error message is returned when you try to create the logical volume.

The following figure illustrates a minimum inter-disk allocation policy with differing Strict settings:

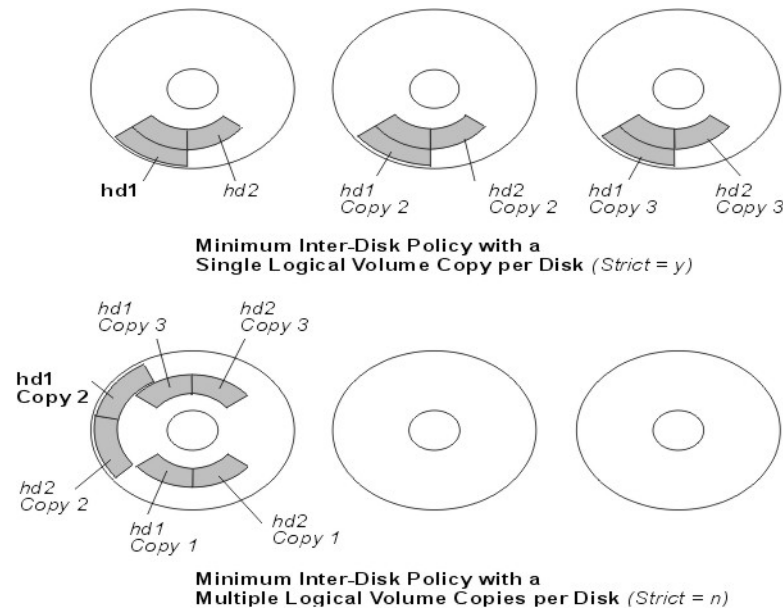


Figure 6. Minimum Inter-Disk Policy/Strict

The following figure illustrates a maximum inter-disk allocation policy with differing Strict settings:

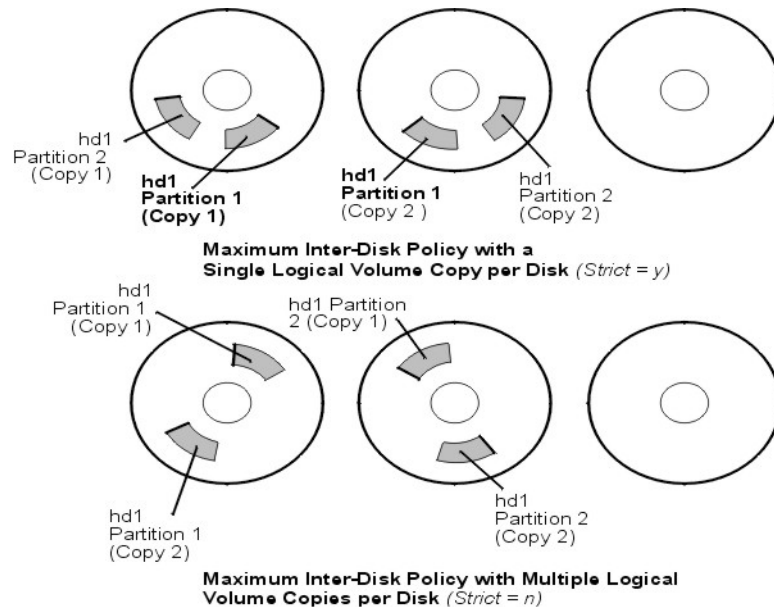


Figure 7. Maximum Inter-Disk Policy/Strict

Related concepts

Inter-disk settings for a single copy of the logical volume

If you select the minimum inter-disk setting (Range = minimum), the physical partitions assigned to the logical volume are located on a single disk to enhance availability. If you select the maximum

inter-disk setting (Range = maximum), the physical partitions are located on multiple disks to enhance performance.

Intra-disk allocation policies for each logical volume

The intra-disk allocation policy choices are based on the five regions of a disk where physical partitions can be located.

The closer a given physical partition is to the center of a physical volume, the lower the average seek time is because the center has the shortest average seek distance from any other part of the disk.

The file system log is a good candidate for allocation at the center of a physical volume because it is so frequently used by the operating system. At the other extreme, the boot logical volume is used infrequently and therefore is allocated at the edge or middle of the physical volume.

The general rule is that the more I/Os, either absolutely or during the running of an important application, the closer to the center of the physical volumes the physical partitions of the logical volume needs to be allocated.

This rule has one important exception: Mirrored logical volumes with Mirror Write Consistency (MWC) set to On are at the outer edge because that is where the system writes MWC data. If mirroring is not in effect, MWC does not apply and does not affect performance.

The five regions where physical partitions can be located are as follows:

1. outer edge
2. inner edge
3. outer middle
4. inner middle
5. center

The edge partitions have the slowest average seek times, which generally result in longer response times for any application that uses them. The center partitions have the fastest average seek times, which generally result in the best response time for any application that uses them. There are, however, fewer partitions on a physical volume at the center than at the other regions.

Combining allocation policies

If you select inter-disk and intra-disk policies that are not compatible, you might get unpredictable results.

The system assigns physical partitions by allowing one policy to take precedence over the other. For example, if you choose an intra-disk policy of center and an inter-disk policy of minimum, the inter-disk policy takes precedence. The system places all of the partitions for the logical volume on one disk if possible, even if the partitions do not all fit into the center region. Make sure you understand the interaction of the policies you choose before implementing them.

Using map files for precise allocation

If the default options provided by the inter- and intra-disk policies are not sufficient for your needs, consider creating map files to specify the exact order and location of the physical partitions for a logical volume.

You can use SMIT, or the **mk1v -m** command to create map files.

For example, to create a ten-partition logical volume called lv06 in the rootvg in partitions 1 through 3, 41 through 45, and 50 through 60 of hdisk1, you could use the following procedure from the command line.

1. To verify that the physical partitions you plan to use are free to be allocated, type:

```
lspv -p hdisk1
```

2. Create a file, such as /tmp/mymap1, containing:

```
hdisk1:1-3  
hdisk1:41-45  
hdisk1:50-60
```

The **mk1v** command allocates the physical partitions in the order that they appear in the map file. Be sure that there are sufficient physical partitions in the map file to allocate the entire logical volume that you specify with the **mk1v** command. (You can list more than you need.)

3. Type the command:

```
mk1v -t jfs -y lv06 -m /tmp/mymap1 rootvg 10
```

Developing striped logical volume strategies

Striped logical volumes are used for large sequential file systems that are frequently accessed and performance-sensitive. Striping is intended to improve performance.

Note: A dump space or boot logical volume cannot be striped. The boot logical volume must be contiguous physical partitions.

To create a 12-partition striped logical volume called lv07 in VGName with a strip size (the strip size multiplied by the number of disks in an array equals the stripe size) of 16 KB across hdisk1, hdisk2, and hdisk3, type:

```
mk1v -y lv07 -S 16K VGName 12 hdisk1 hdisk2 hdisk3
```

To create a 12-partition striped logical volume called lv08 in VGName with a strip size of 8 KB across any three disks within VGName, type:

```
mk1v -y lv08 -S 8K -u 3 VGName 12
```

For more information on how to improve performance by using disk striping, see *Performance management*.

Write-verify policies

Using the write-verify option causes all Write operations to be verified by an immediate follow-up Read operation to check the success of the Write.

If the Write operation is not successful, you get an error message. This policy enhances availability but degrades performance because of the extra time needed for the Read. You can specify the use of a write-verify policy on a logical volume either when you create it using the **mk1v** command, or later by changing it with the **ch1v** command.

Hot-spare disk policies

You can designate disks as hot-spare disks for a volume group with mirrored logical volumes.

When you designate which disks to use as hot-spare disks, you can specify a policy to be used if a disk or disks start to fail and you can specify synchronization characteristics.

If you add a physical volume to a volume group (to mark it as a hot-spare disk), the disk must have at least the same capacity as the smallest disk already in the volume group. When this feature is implemented, data will be migrated to a hot-spare disk when Mirror Write Consistency (MWC) write failures mark a physical volume missing.

The commands to enable hot-spare disk support, **chvg** and **chpv**, provide several options in how you implement the feature at your site, as shown by the following syntax:

```
chvg -hhotsparepolicy -ssyncpolicy VolumeGroup
```

Where *hotsparepolicy* determines which of the following policies you want to use when a disk is failing:

y

Automatically migrates partitions from one failing disk to one spare disk. From the pool of hot spare disks, the smallest one that is big enough to substitute for the failing disk will be used.

Y

Automatically migrates partitions from a failing disk, but might use the complete pool of hot-spare disks.

n

Does not migrate automatically (default).

r

Removes all disks from the pool of hot-spare disks for this volume group.

The *syncpolicy* argument determines whether you want to synchronize any stale partitions automatically:

y

Automatically tries to synchronize stale partitions.

n

Does not automatically try to synchronize stale partitions. (This option is the default.)

The *VolumeGroup* argument specifies the name of the associated mirrored volume group.

Hot spot management in logical volumes

You can identify *hot spot* problems with your logical volumes and remedy those problems without interrupting the use of your system.

A hot-spot problem occurs when some of the logical partitions on your disk have so much disk I/O that your system performance noticeably suffers.

The first step toward solving the problem is to identify it. By default, the system does not collect statistics for logical volume use. After you enable the gathering of these statistics, the first time you enter the **lvmstat** command, the system displays the counter values since the previous system reboot. Thereafter, each time you enter the **lvmstat** command, the system displays the difference since the previous **lvmstat** command.

By interpreting the output of the **lvmstat** command, you can identify the logical partitions with the heaviest traffic. If you have several logical partitions with heavy usage on one physical disk and want to balance these across the available disks, you can use the **migratelp** command to move these logical partitions to other physical disks.

In the following example, the gathering of statistics is enabled and the **lvmstat** command is used repeatedly to gather a baseline of statistics:

```
# lvmstat -v rootvg -e
# lvmstat -v rootvg -C
# lvmstat -v rootvg
```

The output is similar to the following:

Logical Volume	iocnt	Kb_read	Kb_wrtn	Kbps
hd8	4	0	16	0.00
paging01	0	0	0	0.00
lv01	0	0	0	0.00
hd1	0	0	0	0.00
hd3	0	0	0	0.00
hd9var	0	0	0	0.00
hd2	0	0	0	0.00
hd4	0	0	0	0.00
hd6	0	0	0	0.00
hd5	0	0	0	0.00

The previous output shows that all counters have been reset to zero. In the following example, data is first copied from the */unix* directory to the */tmp* directory. The **lvmstat** command output reflects the activity for the rootvg:

```
# cp -p /unix /tmp
# lvmstat -v rootvg
```

Logical Volume	iocnt	Kb_read	Kb_wrtn	Kbps
hd3	296	0	6916	0.04
hd8	47	0	188	0.00
hd4	29	0	128	0.00
hd2	16	0	72	0.00
paging01	0	0	0	0.00
lv01	0	0	0	0.00
hd1	0	0	0	0.00
hd9var	0	0	0	0.00
hd6	0	0	0	0.00
hd5	0	0	0	0.00

The output shows activity on the **hd3** logical volume, which is mounted in the /tmp directory, on **hd8**, which is the JFS log logical volume, on **hd4**, which is / (root), on **hd2**, which is the /usr directory, and on **hd9var**, which is the /var directory. The following output provides details for **hd3** and **hd2**:

```
# lvmstat -l hd3
```

Log_part	mirror#	iocnt	Kb_read	Kb_wrtn	Kbps
1	1	299	0	6896	0.04
3	1	4	0	52	0.00
2	1	0	0	0	0.00
4	1	0	0	0	0.00

```
# lvmstat -l hd2
```

Log_part	mirror#	iocnt	Kb_read	Kb_wrtn	Kbps
2	1	9	0	52	0.00
3	1	9	0	36	0.00
7	1	9	0	36	0.00
4	1	4	0	16	0.00
9	1	1	0	4	0.00
14	1	1	0	4	0.00
1	1	0	0	0	0.00

The output for a volume group provides a summary for all the I/O activity of a logical volume. It is separated into the number of I/O requests (**iocnt**), the kilobytes read and written (**Kb_read** and **Kb_wrtn**, respectively), and the transferred data in KB/s (**Kbps**). If you request the information for a logical volume, you receive the same information, but for each logical partition separately. If you have mirrored logical volumes, you receive statistics for each of the mirror volumes. In the previous sample output, several lines for logical partitions without any activity were omitted. The output is always sorted in decreasing order on the **iocnt** column.

The **migratelp** command uses, as parameters, the name of the logical volume, the number of the logical partition (as it is displayed in the **lvmstat** output), and an optional number for a specific mirror copy. If information is omitted, the first mirror copy is used. You must specify the target physical volume for the move; in addition, you can specify a target physical partition number. If successful, the output is similar to the following:

```
# migratelp hd3/1 hdisk1/109
migratelp: Mirror copy 1 of logical partition 1 of logical volume
hd3 migrated to physical partition 109 of hdisk1.
```

After the hot spot feature is enabled, either for a logical volume or a volume group, you can define your reporting and statistics, display your statistics, select logical partitions to migrate, specify the destination physical partition, and verify the information before committing your changes.

Implementing a volume group policy

After you have decided which volume group policies you want to use, analyze your current configuration by typing the **lspv** command on the command line.

The standard configuration provides a single volume group that includes multiple physical volumes attached to the same disk adapter and other supporting hardware. In a standard configuration, the more disks that make up a quorum volume group the better the chance of the quorum remaining when a disk failure occurs. In a nonquorum group, a minimum of two disks must make up the volume group. To implement your volume group policy changes, do the following:

1. Use the **lspv** command output to check your allocated and free physical volumes.
2. Ensure a quorum by adding one or more physical volumes.
3. Change a volume group to nonquorum status
4. Reconfigure the hardware only if necessary to ensure high availability. For instructions, see the service guide for your system.

Paging space and virtual memory

AIX uses virtual memory to address more memory than is physically available in the system.

The management of memory pages in RAM or on disk is handled by the Virtual Memory Manager (VMM). Virtual-memory segments are partitioned in units called *pages*. A *paging space* is a type of logical volume with allocated disk space that stores information which is resident in virtual memory but is not currently being accessed. This logical volume has an attribute type equal to paging, and is usually simply referred to as paging space or *swap space*. When the amount of free RAM in the system is low, programs or data that have not been used recently are moved from memory to paging space to release memory for other activities.

Paging space concepts

A *paging space* is a type of logical volume with allocated disk space that stores information, which is located in virtual memory but is currently not being accessed.

This logical volume has an attribute type equal to paging, and is usually simply referred to as paging space or *swap space*. When the amount of free RAM in the system is low, programs or data that have not been used recently are moved from memory to paging space to release memory for other activities.

Another type of paging space is available that can be accessed through a device that uses an NFS server for paging-space storage. For an NFS client to access this paging space, the NFS server must have a file created and exported to that client. The file size represents the paging space size for the client.

The amount of paging space required depends on the type of activities performed on the system. If paging space runs low, processes can be lost, and if paging space runs out, the system can panic. When a paging-space low condition is detected, define additional paging space.

The logical volume paging space is defined by making a new paging-space logical volume or by increasing the size of existing paging-space logical volumes. To increase the size of an NFS paging space, the file that resides on the server must be increased by the correct actions on the server.

The total space available to the system for paging is the sum of the sizes of all active paging-space logical volumes.

Paging space allocation policies

The *PSALLOC* environment variable determines which paging space allocation algorithm is used: deferred or early.

AIX uses two modes for paging space allocation. The default is deferred. You can switch to an early paging space allocation mode by changing the value of the *PSALLOC* environment variable, but there are several factors to consider before making such a change. When using the early allocation algorithm, in a worst-case scenario, it is possible to crash the system by using up all available paging space.

Comparisons of deferred and early paging space allocation

The operating system uses the *PSALLOC* environment variable to determine the mechanism used for memory and paging space allocation.

If the *PSALLOC* environment variable is not set, is set to null, or is set to any value other than *early*, the system uses the default *deferred* allocation algorithm.

The *deferred* allocation algorithm helps the efficient use of disk resources and supports applications that prefer a sparse allocation algorithm for resource management. This algorithm does not reserve

paging space when a memory request is made; the disk block allocation of paging space is delayed until it is necessary to page out the requested page. Some programs allocate large amounts of virtual memory and then use only a fraction of the memory. Examples of such programs are technical applications that use sparse vectors or matrices as data structures. The deferred allocation algorithm is also more efficient for a real-time, demand-paged kernel such as the one in the operating system.

This paging space might never be used, especially on systems with large real memory where paging is rare. The deferred algorithm delays allocation of paging space until it is necessary to page out the requested page, which results in no wasted paging space allocation. This delayed allocation can result in a case where the deferred algorithm can try to allocate more paging space than the space available to the system. This situation is called an over-commitment of paging space.

In the over-commitment scenario, where paging space becomes exhausted and an attempt is made to allocate a paging space disk block to page out a page, a failure results. The operating system attempts to avoid complete system failure by killing processes affected by the paging space over-commitment. The **SIGDANGER** signal is sent to notify processes that the amount of free paging space is low. If the paging space situation reaches an even more critical state, selected processes that did not receive the **SIGDANGER** signal are sent a **SIGKILL** signal.

You can use the *PSALLOC* environment variable to switch to an early allocation algorithm, which allocates paging space for the executing process at the time the memory is requested. If there is insufficient paging space available at the time of the request, the early allocation mechanism fails the memory request.

If the *PSALLOC* environment variable is set to *early*, then every program started in that environment from that point on, but not including currently running processes, runs in the early allocation environment. In the early allocation environment, interfaces such as the **malloc** subroutine and the **brk** subroutine will fail if sufficient paging space cannot be reserved when the request is made.

Processes run in the early allocation environment mode are not sent the **SIGKILL** signal if a low paging space condition occurs.

There are different ways to change the *PSALLOC* environment variable to *early*, depending on how broadly you want to apply the change.

The following memory allocation interface subroutines are affected by a switch to an early allocation environment:

- **malloc**
- **free**
- **calloc**
- **realloc**
- **brk**
- **sbrk**
- **shmget**
- **shmctl**

Related tasks

[Configuring the PSALLOC environment variable for early allocation mode](#)

The operating system uses the *PSALLOC* environment variable to determine the mechanism used for memory and paging space allocation.

Early allocation mode

The early allocation algorithm guarantees as much paging space as requested by a memory allocation request. Thus, correct paging space allocation on the system disk is important for efficient operations.

When available paging space drops below a certain threshold, new processes cannot be started and currently running processes might not be able to get more memory. Any processes running under the default deferred allocation mode become highly vulnerable to the **SIGKILL** signal mechanism. In

addition, because the operating system kernel sometimes requires memory allocation, it is possible to crash the system by using up all available paging space.

Before you use the early allocation mode throughout the system, it is very important to define an adequate amount of paging space for the system. The paging space required for early allocation mode is almost always greater than the paging space required for the default deferred allocation mode. How much paging space to define depends on how your system is used and what programs you run. A good starting point for determining the right mix for your system is to define a paging space four times greater than the amount of physical memory.

Certain applications can use extreme amounts of paging space if they are run in early allocation mode. The AIXwindows server currently requires more than 250 MB of paging space when the application runs in early allocation mode. The paging space required for any application depends on how the application is written and how it is run.

All commands and subroutines that show paging space and process memory use include paging space allocated under early allocation mode. The **lsps** command uses the **-s** flag to display total paging space allocation, including paging space allocated under early allocation mode.

Paging space default size

The default paging space size is determined during the system customization phase of AIX installation according to the following standards.

- Paging space can use no less than 16 MB, except for hd6 that can use no less than 64 MB.
- Paging space can use no more than 20% of total disk space.
- If real memory is less than 256 MB, paging space is two times real memory.
- If real memory is greater than or equal to 256 MB, paging space is 512 MB.

Paging space file, commands, and options

The `/etc/swapspaces` file specifies the paging spaces and the attributes of the paging spaces.

A paging space is added to the `/etc/swapspaces` file when it is created by the **mkps** command, and a paging space is removed from the `/etc/swapspaces` file when it is deleted by the **rmps** command. The paging space attributes in the file are modified by the **chps -a** command or the **chps -c** command. Files using a previous format (where there are no attributes for checksum size and automatic swap-on in the stanzas) continue to be supported. If the paging space size is too large, you can subtract logical partitions from the paging space without rebooting using the **chps -d** command.

The following commands are used to manage paging space:

Item	Description
chps	Changes the attributes of a paging space.
lsps	Displays the characteristics of a paging space.
mkps	Adds an additional paging space. The mkps command uses the mklv command with a specific set of options when creating a paging space logical volume. To create NFS paging spaces, the mkps command uses the mkdev command with a different set of options. For NFS paging spaces, the mkps command needs the host name of the NFS server and the path name of the file that is exported from the server.
rmps	Removes an inactive paging space.
swapoff	Deactivates one or more paging space without rebooting the system. Information in the paging space is moved to other active paging space areas. The deactivated paging space can then be removed using the rmps command.

Item	Description
<u>swapon</u>	Activates a paging space. The swapon command is used during early system initialization to activate the initial paging-space device. During a later phase of initialization, when other devices become available, the swapon command is used to activate additional paging spaces so that paging activity occurs across several devices.

The `paging` type option is required for all logical volume paging spaces.

The following options are used to maximize paging performance with a logical volume:

- Allocate in the middle of the disk to reduce disk arm travel
- Use multiple paging spaces, each allocated from a separate physical volume.

Configuring paging space

Many of the configuring tasks can be done with SMIT. Paging space and memory allocation is controlled by the **PSALLOC** environment variable.

Adding and activating paging space

To make paging space available to your system, you must add and activate the paging space.

The total amount of paging space is often determined by trial and error. One commonly used guideline is to double the RAM size and use that figure as a paging space target.

Using the SMIT interface, type one of the following fast paths on the command line:

- To list your current paging space, type: `smit lsp`
- To add paging space, type: `smit mkps`
- To activate paging space, type: `smit swapon`

Improving paging performance

To improve paging performance, use multiple paging spaces and locate them on separate physical volumes whenever possible.

However, more than one paging space can be located on the same physical volume. Although you can use multiple physical volumes, it is a good idea to select only those disks within the rootvg volume group unless you are thoroughly familiar with your system.

Configuring the PSALLOC environment variable for early allocation mode

The operating system uses the *PSALLOC* environment variable to determine the mechanism used for memory and paging space allocation.

The default setting is `late`. The following examples show different ways to change the *PSALLOC* environment variable to `early`. The method you choose depends on how broadly you want to apply the change.

- Type the following command on a shell command line:

```
PSALLOC=early;export PSALLOC
```

This command causes all subsequent commands run from that shell session to run in early allocation mode.

- Add the following command in a shell resource file (`.shrc` or `.kshrc`):

```
PSALLOC=early;export PSALLOC
```

This entry causes all processes in your login session, with the exception of the login shell, to run under early allocation mode. This method also protects the processes from the **SIGKILL** signal mechanism.

- Insert the **putenv** subroutine inside a program to set the *PSALLOC* environment variable to *early*. Using this method, the early allocation behavior takes effect at the next call to the **exec** subroutine.

Related concepts

Comparisons of deferred and early paging space allocation

The operating system uses the *PSALLOC* environment variable to determine the mechanism used for memory and paging space allocation.

Changing or removing a paging space

Changing a paging space is easily done with SMIT, but removing a paging space is more risky.

Changing the characteristics of a paging space can be done with the following SMIT fast path on the command line: `smit chps`.

The procedure to remove a paging space is more risky, especially if the paging space you want to remove is a default paging space, such as `hd6`. A special procedure is required for removing the default paging spaces, because they are activated during boot time by shell scripts that configure the system. To remove one of the default paging spaces, these scripts must be altered and a new boot image must be created.



Attention: Removing default paging spaces incorrectly can prevent the system from restarting. The following procedure is for experienced system managers only.

To remove an existing paging space, use the following procedure:

1. With root authority, deactivate the paging space by typing the following SMIT fast path on the command line:

```
smit swapoff
```

2. If the paging space you are removing is the default dump device, you must change the default dump device to another paging space or logical volume before removing the paging space.

To change the default dump device, type the following command:

```
sysdumpdev -P -p /dev/new_dump_device
```

3. Remove the paging space by typing the following fast path:

```
smit rmpps
```

Using the paging space allocation mode programming interface

The programming interface that controls the paging space allocation mode uses the *PSALLOC* environment variable.

To ensure that an application always runs under the desired mode (with or without early paging space allocation), do the following:

1. Use the **getenv** subroutine to examine the current state of the *PSALLOC* environment variable.
2. If the value of the *PSALLOC* environment variable is not the value required by the application, use the **setenv** subroutine to alter the value of the environment variable.

Because only the **execve** subroutine examines the state of the *PSALLOC* environment variable, call the **execve** subroutine with the same set of parameters and environment received by the application. When the application reexamines the state of the *PSALLOC* environment variable and finds the correct value, the application continues normally.

3. If the **getenv** subroutine reveals that the current state of the *PSALLOC* environment variable is correct, no modification is needed.

The application continues normally.

Relocating and reducing hd6 paging space

You might want to reduce or move the default paging space in order to enhance storage system performance by forcing paging and swapping to other disks in the system that are less busy. Reducing or moving the default paging also conserves disk space on hdisk0.

Whether moving the paging space or reducing its size, the rationale is the same: move paging space activity to disks that are less busy. The installation default creates a paging logical volume (hd6) on drive hdisk0, that contains part or all of the busy / (root) and /usr file systems. If the minimum inter-disk allocation policy is chosen, meaning that all of / and a large amount of /usr are on hdisk0, moving the paging space to a disk that is less busy can significantly improve performance. Even if the maximum inter-disk allocation policy is implemented and both / and /usr are distributed across multiple physical volumes, your hdisk2 (assuming three disks) likely contains fewer logical partitions belonging to the busiest file systems.

The following procedures describe how to make the hd6 paging space smaller and how to move the hd6 paging space within the same volume group.

Making the hd6 paging space smaller

The following procedure uses the **chps** command to shrink existing paging spaces, including the primary paging space and the primary and secondary dump device.

The **chps** command calls the **shrinkps** script, which safely shrinks the paging space without leaving the system in an unbootable state. Specifically, the script does the following:

1. Creates a temporary paging space in the same volume
2. Moves information to that temporary space
3. Creates a new, smaller paging space in the same volume
4. Removes the old paging space

For the **chps** command to complete successfully, enough free disk space (space not allocated to any logical volume) must exist to create a temporary paging space. The size of the temporary paging space is equal to amount of space needed to hold all the paged out pages in the old paging space. The minimum size for a primary paging space is 32 MB. The minimum size for any other paging space is 16 MB.

Note: If the following procedure encounters an I/O error, the system might require immediate shutdown and rebooting.

1. Check your logical volume and file system distribution across physical volumes by typing the following command:

```
lspv -l hdiskX
```

Where *hdiskX* is the name of your physical volume.

2. To shrink the paging space size, type the following on the command line:

```
smit chps
```

Note: The primary paging space is hardcoded in the boot record. Therefore, the primary paging space will always be activated when the system is restarted. The **chps** command cannot deactivate the primary paging space.

Priority is given to maintaining an operational configuration. System checks can lead to immediate refusal to shrink the paging space. Errors occurring while the temporary paging space is being created cause the procedure to exit, and the system will revert to the original settings. Other problems are likely to provoke situations that will require intervention by the system administrator or possibly an immediate reboot. Some errors may prevent removal of the temporary paging space. This would normally require non-urgent attention from the administrator.



Attention: If an I/O error is detected on system backing pages or user backing pages by the **swapoff** command within the **shrinkps** script, an immediate shutdown is advised to avoid a possible system crash. At reboot, the temporary paging space is active and an attempt can be

made to stop and restart the applications which encountered the I/O errors. If the attempt is successful and the **swapoff** command is able to complete deactivation, the shrink procedure can be completed manually using the **mkps**, **swapoff** and **rmps** commands to create a paging space with the required size and to remove the temporary paging space.

Do not attempt to remove (using **rmps**) or reactivate (using **chps**) a deactivated paging space that was in the I/O ERROR state before the system restart. There is a risk that the disk space will be reused and may cause additional problems.

Moving the hd6 paging space within the same volume group

Moving the default paging space from `hdisk0` to a different disk within the same volume group does not require the system to shut down and reboot.

With root authority, type the following command to move the default (`hd6`) paging space from `hdisk0` to `hdisk2`:

```
migratepv -l hd6 hdisk0 hdisk2
```



Attention: Moving a paging space with the name `hd6` from `rootvg` to another volume group is not recommended because the name is hardcoded in several places, including the second phase of the boot process and the process that accesses the root volume group when booting from removable media. Only the paging spaces in `rootvg` are active during the second phase of the boot process, and having no paging space in `rootvg` could severely affect system boot performance. If you want the majority of paging space on other volume groups, it is better to make `hd6` as small as possible (the same size as physical memory) and then create larger paging spaces on other volume groups..

Troubleshooting paging space

The most common problem regarding paging space is caused by running out of allocated space.

The total amount of paging space is often determined by trial and error. One commonly used guideline is to double the RAM size and use that figure as a paging space target. If paging space runs low, processes can be lost, and if paging space runs out, the system can panic. The following signal and error information can help you monitor and resolve or prevent paging space problems.

The operating system monitors the number of free paging space blocks and detects when a paging-space shortage exists. When the number of free paging-space blocks falls below a threshold known as the *paging-space warning level*, the system informs all processes (except **kprocs**) of this condition by sending the **SIGDANGER** signal. If the shortage continues and falls below a second threshold known as the *paging-space kill level*, the system sends the **SIGKILL** signal to processes that are the major users of paging space and that do not have a signal handler for the **SIGDANGER** signal. (The default action for the **SIGDANGER** signal is to ignore the signal.) The system continues sending **SIGKILL** signals until the number of free paging-space blocks is above the paging-space kill level.

Note: If the `low_ps_handling` parameter is set to 2 (under the **vmo** command) and if no process was found to kill (without the **SIGDANGER** handler), the system will send the **SIGKILL** signal to the youngest processes that have a signal handler for the **SIGDANGER** signal.

Processes that dynamically allocate memory can ensure that sufficient paging space exists by monitoring the paging-space levels with the **psdanger** subroutine or by using special allocation routines. You can use the **disclaim** subroutine to prevent processes from ending when the paging-space kill level is reached. To do this, define a signal handler for the **SIGDANGER** signal and release memory and paging-space resources allocated in their data and stack areas and in shared memory segments.

If you get error messages similar to the following, increase the paging space:

```
INIT: Paging space is low!
```

OR

```
You are close to running out of paging space.  
You may want to save your documents because
```



```
this program (and possibly the operating system)
could terminate without future warning when the
paging space fills up.
```

Virtual Memory Manager

The Virtual Memory Manager (VMM) manages the memory requests made by the system and its applications.

Virtual-memory segments are partitioned in units called *pages*; each page is either located in real physical memory (RAM) or stored on disk until it is needed. AIX uses virtual memory to address more memory than is physically available in the system. The management of memory pages in RAM or on disk is handled by the VMM.

Real memory management in Virtual Memory Manager

In AIX, virtual-memory segments are partitioned into 4096-byte units called pages. Real memory is divided into 4096-byte page frames.

The VMM has two major functions:

- Manage the allocation of page frames
- Resolve references to virtual-memory pages that are not currently in RAM (stored in paging space) or do not yet exist.

To accomplish these functions, the VMM maintains a *free list* of available page frames. The VMM also uses a page-replacement algorithm to determine which virtual-memory pages currently in RAM will have their page frames reassigned to the free list. The page-replacement algorithm takes into account the existence of persistent versus working segments, repaging, and VMM thresholds.

Virtual Memory Manager Free list

The VMM maintains a list of free (unallocated) page frames that it uses to satisfy page faults.

AIX tries to use all of RAM all of the time, except for a small amount which it maintains on the free list. To maintain this small amount of unallocated pages the VMM uses *page outs* and *page steals* to free up space and reassign those page frames to the free list. The virtual-memory pages whose page frames are to be reassigned are selected using the VMM's page-replacement algorithm.

Persistent or working memory segments in Virtual Memory Manager

AIX distinguishes between different types of memory segments. To understand the VMM, it is important to understand the difference between working and persistent segments.

A *persistent segment* has a permanent storage location on disk. Files containing data or executable programs are mapped to persistent segments. When a JFS or JFS2 file is opened and accessed, the file data is copied into RAM. VMM parameters control when physical memory frames allocated to persistent pages should be overwritten and used to store other data.

Working segments are transitory and exist only during their use by a process. Working segments have no permanent disk storage location. Process stack and data regions are mapped to working segments and shared library text segments. Pages of working segments must also occupy disk storage locations when they cannot be kept in real memory. The disk paging space is used for this purpose. When a program exits, all of its working pages are placed back on the free list immediately.

Working segments and paging space in Virtual Memory Manager

Working pages in RAM that can be modified and paged out are assigned a corresponding slot in paging space.

The allocated paging space is used only if the page needs to be paged out. However, an allocated page in paging space cannot be used by another page. It remains reserved for a particular page for as long as that

page exists in virtual memory. Because persistent pages are paged out to the same location on disk from which they came, paging space does not need to be allocated for persistent pages residing in RAM.

The VMM has two modes for allocating paging space: *early* and *late*. Early allocation policy reserves paging space whenever a memory request for a working page is made. Late allocation policy only assigns paging space when the working page is actually paged out of memory, which significantly reduces the paging space requirements of the system.

Virtual Memory Manager memory load control facility

When a process references a virtual-memory page that is on disk, because it either has been paged out or has never been read, the referenced page must be paged in, and this might cause one or more pages to be paged out if the number of available (free) page frames is low. The VMM attempts to steal page frames that have not been recently referenced and, therefore, are not likely to be referenced in the near future, using a page-replacement algorithm.

A successful page-replacement keeps the memory pages of all currently active processes in RAM, while the memory pages of inactive processes are paged out. However, when RAM is over-committed, it becomes difficult to choose pages for page out because, they will probably be referenced in the near future by currently running processes. The result is that pages that are likely to be referenced soon might still get paged out and then paged in again when actually referenced. When RAM is over-committed, continuous paging in and paging out, called *thrashing*, can occur. When a system is thrashing, the system spends most of its time paging in and paging out instead of executing useful instructions, and none of the active processes make any significant progress. The VMM has a memory load control algorithm that detects when the system is thrashing and then attempts to correct the condition.

File systems

A *file system* is a hierarchical structure (file tree) of files and directories.

This type of structure resembles an inverted tree with the roots at the top and branches at the bottom. This file tree uses directories to organize data and programs into groups, allowing the management of several directories and files at one time.

A file system resides on a single logical volume. Every file and directory belongs to a file system within a logical volume. Because of its structure, some tasks are performed more efficiently on a file system than on each directory within the file system. For example, you can back up, move, or secure an entire file system. You can make an point-in-time image of a JFS file system or a JFS2 file system, called a *snapshot*.

Note: The maximum number of logical partitions per logical volume is 32,512. For more information on file system logical volume characteristics, see the [chlv](#) command.

The [mkfs](#) (make file system) command or the System Management Interface Tool ([smit](#) command) creates a file system on a logical volume.

To be accessible, a file system must be mounted onto a directory mount point. When multiple file systems are mounted, a directory structure is created that presents the image of a single file system. It is a hierarchical structure with a single root. This structure includes the base file systems and any file systems you create. You can access both local and remote file systems using the [mount](#) command. This makes the file system available for read and write access from your system. Mounting or unmounting a file system usually requires system group membership. File systems can be mounted automatically, if they are defined in the [/etc/filesystems](#) file. You can unmount a local or remote file system with the [umount](#) command, unless a user or process is accessing that file system. For more information on mounting a file system, see [“Mounting”](#) on page 100.

The basic type of file system used by AIX is called the *journaled file system (JFS)*. This file system uses database journaling techniques to maintain its structural consistency. This prevents damage to the file system when the system is halted abnormally.

The AIX operating system supports multiple file system types including the journaled file system (JFS) and the enhanced journaled file system (JFS2). For more information on file system types and the characteristics of each type, see [“File system types”](#) on page 105.

Some of the most important system management tasks have to do with file systems, specifically:

- Allocating space for file systems on logical volumes
- Creating file systems
- Making file system space available to system users
- Monitoring file system space usage
- Backing up file systems to guard against data loss if the system fails
- Maintaining file systems in a consistent state

These tasks should be performed by your system administrator.

File system concepts

Before you can manage and configure your file system you need to understand the basic organization and content of the file tree.

Organization and contents of the file tree

The file tree organizes files into directories containing similar information. This organization facilitates remote mounting of directories and files.

System administrators can use these directories as building blocks to construct a unique file tree for each client mounting individual directories from one or more servers. Mounting files and directories remotely, rather than keeping all information local, has the following advantages:

- Conserves disk space
- Allows easy, centralized system administration
- Provides a more secure environment

The file tree has the following characteristics:

- Files that can be shared by machines of the same hardware architecture are located in the `/usr` file system.
- Variable per-client files, such as spool and mail files, are located in the `/var` file system.
- Architecture-independent, shareable text files, such as manual pages, are located in the `/usr/share` directory.
- The `/` (root) file system contains files and directories critical for system operation. For example, it contains a device directory, programs used for system startup, and mount points where file systems can be mounted onto the root file system.
- The `/home` file system is the mount point for user home directories.

File system structure

It is important to understand the difference between a file system and a directory. A file system is a section of hard disk that has been allocated to contain files. This section of hard disk is accessed by mounting the file system over a directory. After the file system is mounted, it looks just like any other directory to the end user.

However, because of the structural differences between the file systems and directories, the data within these entities can be managed separately.

When the operating system is installed for the first time, it is loaded into a directory structure, as shown in the following illustration.

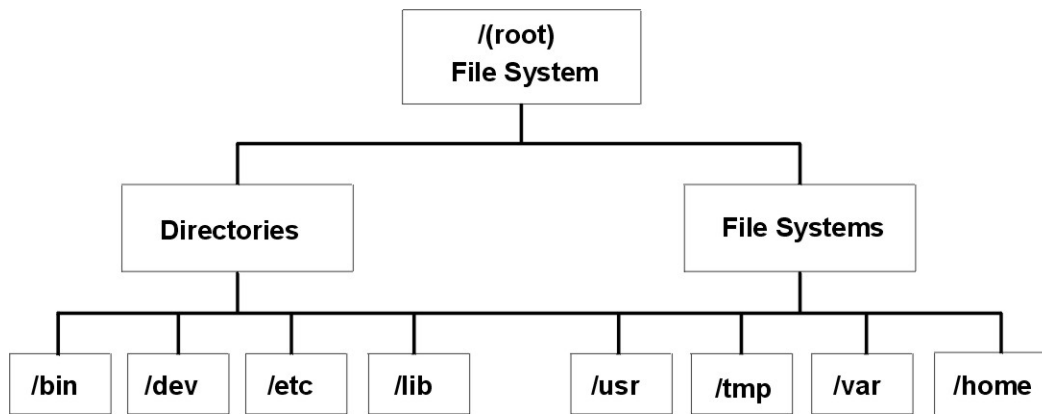


Figure 8. / (root) File System Tree

The directories on the right (/usr, /tmp, /var, and /home) are all file systems so they have separate sections of the hard disk allocated for their use. These file systems are mounted automatically when the system is started, so the end user does not see the difference between these file systems and the directories listed on the left (/bin, /dev, /etc, and /lib).

On standalone machines, the following file systems reside on the associated device by default:

/Device	/File System
/dev/hd1	/home
/dev/hd2	/usr
/dev/hd3	/tmp
/dev/hd4	/(root)
/dev/hd9var	/var
/proc	/proc
/dev/hd10opt	/opt

The file tree has the following characteristics:

- Files that can be shared by machines of the same hardware architecture are located in the /usr file system.
- Variable per-client files, for example, spool and mail files, are located in the /var file system.
- The /(root) file system contains files and directories critical for system operation. For example, it contains
 - A device directory (/dev)
 - Mount points where file systems can be mounted onto the root file system, for example, /mnt
- The /home file system is the mount point for users' home directories.
- For servers, the /export directory contains paging-space files, per-client (unshared) root file systems, dump, home, and /usr/share directories for diskless clients, as well as exported /usr directories.
- The /proc file system contains information about the state of processes and threads in the system.
- The /opt file system contains optional software, such as applications.

The following list provides information about the contents of some of the subdirectories of the /(root) file system.

Item	Description
/bin	<u>Symbolic link</u> to the /usr/bin directory.

Item	Description
/dev	Contains device nodes for special files for local devices. The /dev directory contains special files for tape drives, printers, disk partitions, and terminals.
/etc	Contains configuration files that vary for each machine. Examples include: <ul style="list-style-type: none"> • /etc/hosts • /etc/passwd
/export	Contains the directories and files on a server that are for remote clients.
/home	Serves as a mount point for a file system containing user home directories. The /home file system contains per-user files and directories. <p>In a standalone machine, a separate local file system is mounted over the /home directory. In a network, a server might contain user files that should be accessible from several machines. In this case, the server's copy of the /home directory is remotely mounted onto a local /home file system.</p>
/lib	Symbolic link to the /usr/lib directory, which contains architecture-independent libraries with names in the form lib*.a.
/sbin	Contains files needed to boot the machine and mount the /usr file system. Most of the commands used during booting come from the boot image's RAM disk file system; therefore, very few commands reside in the /sbin directory.
/tmp	Serves as a mount point for a file system that contains system-generated temporary files.
/u	Symbolic link to the /home directory.
/usr	Serves as a mount point for a file system containing files that do not change and can be shared by machines (such as executable programs and ASCII documentation). <p>Standalone machines mount a separate local file system over the /usr directory. Diskless and disk-poor machines mount a directory from a remote server over the /usr file system.</p>
/var	Serves as a mount point for files that vary on each machine. The /var file system is configured as a file system because the files that it contains tend to grow. For example, it is a symbolic link to the /usr/tmp directory, which contains temporary work files.

Root file system

The root file system is the top of the hierarchical file tree. It contains the files and directories critical for system operation, including the device directory and programs for booting the system. The root file system also contains mount points where file systems can be mounted to connect to the root file system hierarchy.

The following diagram shows many of the subdirectories of the root file system.

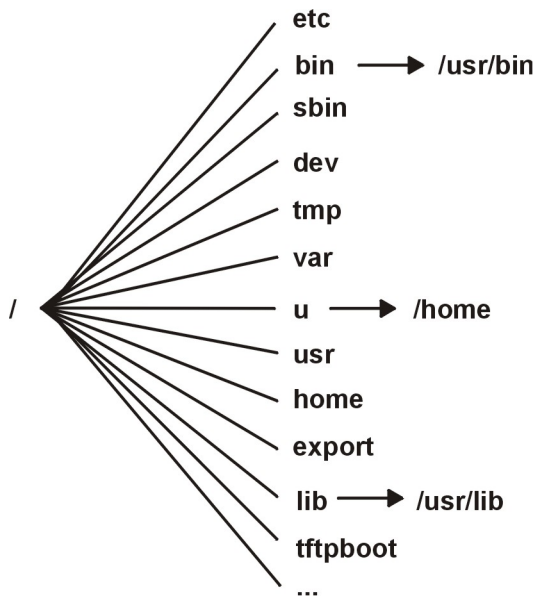


Figure 9. Root File System

The following list provides information about the contents of some of the subdirectories of the / (root) file system.

Item	Description
/etc	<p>Contains configuration files that vary for each machine. Examples include:</p> <ul style="list-style-type: none"> • /etc/hosts • /etc/passwd <p>The /etc directory contains the files generally used in system administration. Most of the commands that previously resided in the /etc directory now reside in the /usr/sbin directory. However, for compatibility, the /usr/sbin directory contains symbolic links to the locations of some executable files. Examples include:</p> <ul style="list-style-type: none"> • /etc/chown is a symbolic link to /usr/bin/chown. • /etc/exportvg is a symbolic link to /usr/sbin/exportvg.
/bin	Symbolic link to the /usr/bin directory. In prior UNIX file systems, the /bin directory contained user commands that now reside in the /usr/bin directory.
/sbin	Contains files needed to boot the machine and mount the /usr file system. Most of the commands used during booting come from the boot image's RAM disk file system; therefore, very few commands reside in the /sbin directory.
/dev	Contains device nodes for special files for local devices. The /dev directory contains special files for tape drives, printers, disk partitions, and terminals.
/tmp	Serves as a mount point for a file system that contains system-generated temporary files. The /tmp file system is an empty directory.
/var	Serves as a mount point for files that vary on each machine. The /var file system is configured as a file system since the files it contains tend to grow. See “/var file system” on page 80 for more information.
/u	Symbolic link to the /home directory.

Item	Description
/usr	<p>Contains files that do not change and can be shared by machines such as executables and ASCII documentation.</p> <p>Standalone machines mount the root of a separate local file system over the /usr directory. Diskless machines and machines with limited disk resources mount a directory from a remote server over the /usr file system. See “/usr file system” on page 78 for more information about the file tree mounted over the /usr directory.</p>
/home	<p>Serves as a mount point for a file system containing user home directories. The /home file system contains per-user files and directories.</p> <p>In a standalone machine, the /home directory is contained in a separate file system whose root is mounted over the /home directory root file system. In a network, a server might contain user files that are accessible from several machines. In this case, the server copy of the /home directory is remotely mounted onto a local /home file system.</p>
/export	<p>Contains the directories and files on a server that are for remote clients.</p> <p>See “/export directory” on page 81 for more information about the file tree that resides under the /export directory.</p>
/lib	Symbolic link to the /usr/lib directory. See “/usr file system” on page 78 for more information.
/tftpboot	Contains boot images and boot information for diskless clients.

/usr file system

The /usr file system contains executable files that can be shared among machines.

The major subdirectories of the /usr directory are shown in the following diagram.

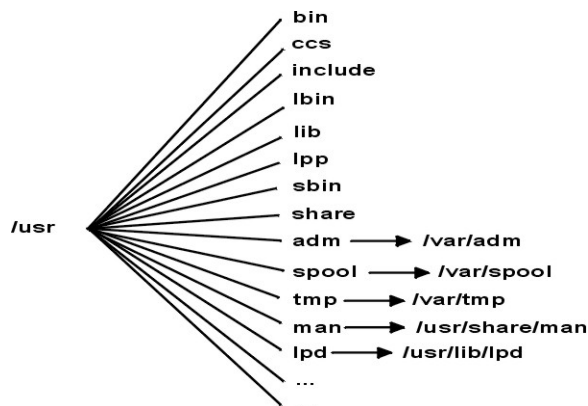


Figure 10. /usr File System

On a standalone machine the /usr file system is a separate file system (in the /dev/hd2 logical volume). On a diskless machine or a machine with limited disk resources, a directory from a remote server is mounted with read-only permissions over the local /usr file system. The /usr file system contains read-only commands, libraries, and data.

Except for the contents of the /usr/share directory, the files and directories in the /usr file system can be shared by all machines of the same hardware architecture.

The /usr file system includes the following directories:

Item	Description
/usr/bin	Contains ordinary commands and shell scripts. For example, the /usr/bin directory contains the ls , cat , and mkdir commands.

Item	Description
<code>/usr/ccs</code>	Contains unbundled development package binaries.
<code>/usr/include</code>	Contains include, or header, files.
<code>/usr/lbin</code>	Contains executable files that are backends to commands.
<code>/usr/lib</code>	Contains architecture-independent libraries with names of the form <code>lib*.a</code> . The <code>/lib</code> directory in <code>/</code> (root) is a symbolic link to the <code>/usr/lib</code> directory, so all files that were once in the <code>/lib</code> directory are now in the <code>/usr/lib</code> directory. This includes a few nonlibrary files for compatibility.
<code>/usr/lpp</code>	Contains optionally installed products.
<code>/usr/sbin</code>	Contains utilities used in system administration, including System Management Interface Tool (SMIT) commands. Most of the commands that once resided in the <code>/etc</code> directory now reside in the <code>/usr/sbin</code> directory.
<code>/usr/share</code>	Contains files that can be shared among machines with different architectures. See “/usr/share directory” on page 79 for more information.

The following are symbolic links to the `/var` directory:

Item	Description
<code>/usr/adm</code>	Symbolic link to the <code>/var/adm</code> directory
<code>/usr/mail</code>	Symbolic link to the <code>/var/spool/mail</code> directory
<code>/usr/news</code>	Symbolic link to the <code>/var/news</code> directory
<code>/usr/preserve</code>	Symbolic link to the <code>/var/preserve</code> directory
<code>/usr/spool</code>	Symbolic link to the <code>/var/spool</code> directory
<code>/usr/tmp</code>	Symbolic link to the <code>/var/tmp</code> directory, because the <code>/usr</code> directory is potentially shared by many nodes and is read-only

The following are symbolic links to the `/usr/share` and `/usr/lib` directories:

Item	Description
<code>/usr/dict</code>	Symbolic link to the <code>/usr/share/dict</code> directory
<code>/usr/man</code>	Symbolic link to the <code>/usr/share/man</code> directory
<code>/usr/lpd</code>	Symbolic link to the <code>/usr/lib/lpd</code> directory

/usr/share directory

The `/usr/share` directory contains architecture-independent shareable text files. The contents of this directory can be shared by all machines, regardless of hardware architecture.

In a mixed architecture environment, the typical diskless client mounts one server directory over its own `/usr` directory and then mounts a different directory over the `/usr/share` directory. The files below the `/usr/share` directory are contained in one or more separately installable packages. Thus, a node might have the other parts of the `/usr` directory it depends on locally installed while using a server to provide the `/usr/share` directory.

Some of the files in the `/usr/share` directory include the directories and files shown in the following diagram.

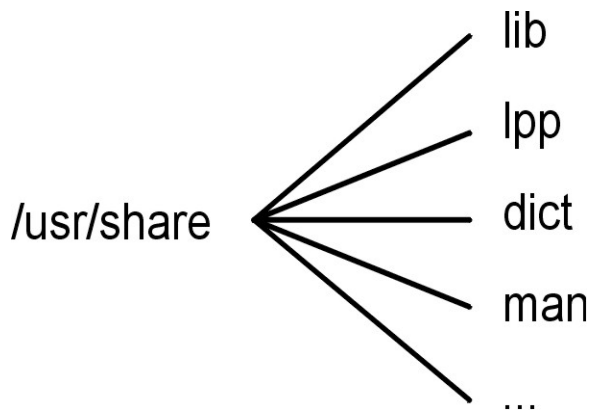


Figure 11. /usr/share Directory

This diagram shows several directories under the /usr/share directory, including /lib, /lpp, /dict, and /man.

The /usr/share directory includes the following:

Item	Description
/usr/share/man	Contains the manual pages if they have been loaded
/usr/share/dict	Contains the spelling dictionary and its indexes
/usr/share/lib	Contains architecture-independent data files, including terminfo, learn, tmac, me, and macros
/usr/share/lpp	Contains data and information about optionally installable products on the system

/var file system

The /var file system tends to grow because it contains subdirectories and data files that are used by busy applications such as accounting, mail, and the print spooler.



Attention: If applications on your system use the /var file system extensively, routinely run the **skulker** command or increase the file system size beyond the 4MB /var default.

Specific /var files that warrant periodic monitoring are /var/adm/wtmp and /var/adm/ras/errlog.

Other /var files to monitor are:

Item	Description
/var/adm/ras/trcfile	If the trace facility is turned on
/var/tmp/snmpd.log	If the snmpd command is running on your system

The /var directory diagram shows some of the directories in the /var file system.

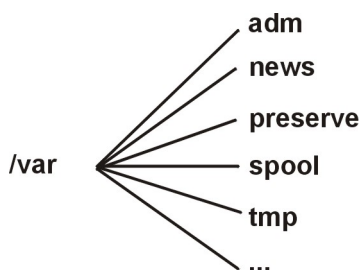


Figure 12. /var Directory

Item	Description
<code>/var/adm</code>	Contains system logging and accounting files
<code>/var/news</code>	Contains system news
<code>/var/preserve</code>	Contains preserved data from interrupted edit sessions; similar to the <code>/usr/preserve</code> directory in previous releases
<code>/var/spool</code>	Contains files being processed by programs such as electronic mail; similar to the <code>/usr/spool</code> directory in previous releases
<code>/var/tmp</code>	Contains temporary files; similar to the <code>/usr/tmp</code> directory in previous releases. The <code>/usr/tmp</code> directory is now a symbolic link to <code>/var/tmp</code> .

/export directory

The `/export` directory contains server files exported to clients, such as diskless, dataless, or disk-poor machines.

A server can export several types of disk space, including packages of executable programs, paging space for diskless clients, and root file systems for diskless clients or those with low disk resources. The standard location for such disk space in the file tree is the `/export` directory. Some of the subdirectories of the `/export` directory are shown in the following list:

/exec

Contains directories that diskless clients mount over their `/usr` file systems

/swap

Contains files for diskless clients' remote paging

/share

Contains directories that diskless clients mount over their `/usr/share` directory

/root

Contains directories that diskless clients mount over their `/ (root)` file system

/home

Contains directories that diskless clients mount over their `/home` file system

The `/export` directory is the default location for client resources for the diskless commands. The `/export` directory is only the location of client resources on the server. Because clients mount these resources onto their own file tree, these resources appear to clients at the normal places in a file tree. The major subdirectories of the `/export` directory, and their corresponding mount points on a client file tree, include:

/export/root

This directory is mounted over the client root (`/`) file system. Client root directories are located in the `/export/root` directory by default and are named with the client's host name.

/export/exec

Also called the Shared Product Object Tree (SPOT) directory. This directory is mounted over the client `/usr` file system. SPOTs are versions of the `/usr` file system stored in the `/export/exec` directory and have names that reflect their release level. By default, the name is `RISCAIX`.

/export/share

This directory is mounted over the client `/usr/share` directory. This directory contains data that can be shared by many architectures. The default location is `/export/share/AIX/usr/share`.

/export/home

This directory is mounted over the client `/home` file system. It contains user directories grouped by client host names. The default location for client home directories is `/export/home`.

/export/swap

Also called the paging directory. In standalone or dataless systems, paging is provided by a local disk; for diskless clients, this service is provided by a file on a server. This file is named after the client's host name and by default is found in the `/export/swap` directory.

/export/dump

Standalone systems use a local disk as the dump device; diskless clients use a file on a server. The file resides in a directory named after the client host name and by default is found in the /export/dump directory.

microcode

This directory contains microcode for physical devices. The default location is /export/exec/RISCAIX/usr/lib/microcode.

Encrypting JFS2 file systems

Beginning with AIX Version 6.1, Encrypted File System (EFS) is supported on JFS2 file systems. EFS allows you to encrypt your data and control access to the data through keyed protection.

A key is associated with each user and is stored in a cryptographically protected key store. Upon successful login, the user's keys are loaded into the kernel and associated with the process credentials. To open an EFS-protected file, the process credentials are tested. If the process finds a key that matches the file protection, the process decrypts the file key and the file content.

By default, JFS2 file systems are not EFS-enabled. A JFS2 file system must be **EFS-enabled** before any data can be encrypted.

Configuring file systems

When adding or configuring file systems, you can select options in the File Systems container of the SMIT fast paths.

The SMIT fast paths are provided in the following table:

<i>Table 4. Managing Logical Volumes and File Systems Tasks</i>	
Task	SMIT Fast Path
Add a JFS or JFS2	smit crfs
Add a JFS2 to an existing logical volume	smit crjfs2lvstd
Add a JFS to a previously defined logical volume menu	Create logical volume, then smit crjfs1v
Change the attributes of a JFS or a JFS2 ^{Note 1}	smit chfs
Check size of a file system	smit fs
Increase size of a file system	JFS: smit chjfs JFS2: smit chjfs2
Decrease size of a file system	JFS2: smit chjfs2

Note: The SMIT Fast Path for **Decrease size of a file system** is only for JFS2.

Managing file system

A file system is a complete directory structure, including a root directory and any subdirectories and files beneath it.

File systems are confined to a single logical volume. Some of the most important system management tasks are concerning file systems, specifically:

- Allocating space for file systems on logical volumes
- Creating file systems
- Making file system space available to system users
- Monitoring file system space usage
- Backing up file systems to guard against data loss in the event of system failures

- Making a snapshot to capture a consistent block-level image of a file system at a given point in time
- Maintaining file systems in a consistent state.

Following is a list of system management commands that help manage file systems:

Item	Description
<u>backup</u>	Performs a full or incremental backup of a file system
<u>chfs -a splitcopy</u>	Creates an online backup of a mounted JFS file system
<u>dd</u>	Copies data directly from one device to another for making file system backups
<u>df</u>	Reports the amount of space used and free on a file system
<u>fsck</u>	Checks file systems and repairs inconsistencies
<u>mkfs</u>	Makes a file system of a specified size on a specified logical volume
<u>mount</u>	Attaches a file system to the system-wide naming structure so that files and directories in that file system can be accessed
<u>restore</u>	Restores files from a backup
<u>snapshot</u>	Creates a snapshot of a JFS2 file system
<u>umount</u>	Removes a file system from the system-wide naming structure, making the files and directories in the file system inaccessible.

Displaying available space on a file system (df command)

Use the **df** command to display information about total space and available space on a file system. The **FileSystem** parameter specifies the name of the device on which the file system resides, the directory on which the file system is mounted, or the relative path name of a file system.

If you do not specify the **FileSystem** parameter, the **df** command displays information for all currently mounted file systems. If a file or directory is specified, then the **df** command displays information for the file system on which it resides.

Normally, the **df** command uses free counts contained in the superblock. Under certain exceptional conditions, these counts might be in error. For example, if a file system is being actively modified when the **df** command is running, the free count might not be accurate.

Note: On some remote file systems, such as Network File Systems (NFS), the columns representing the available space on the display are left blank if the server does not provide the information.

The following are examples of how to use the **df** command:

- To display information about all mounted file systems, type the following:

```
df
```

If your system is configured so the `/`, `/usr`, `/site`, and `/usr/venus` directories reside in separate file systems, the output from the **df** command is similar to the following:

```
Filesystem 512-blocks  free   %used    Iused %Iused   Mounted on
/dev/hd4    20480    13780   32%      805   13%      /
/dev/hd2   385024    15772   95%    27715  28%     /usr
/dev/hd9var 40960    38988    4%      115    1%     /var
/dev/hd3    20480    18972    7%       81    1%     /tmp
/dev/hd1     4096     3724    9%       44    4%     /home
```

- To display available space on the file system in which your current directory resides, type the following:

```
df .
```

File system commands

There are a number of commands designed to operate on file systems, regardless of type.

The `/etc/filesystems` file controls the list of file systems that the following commands can manipulate:

Item	Description
<code>chfs</code>	Changes the characteristics of a file system
<code>crfs</code>	Adds a file system
<code>lsfs</code>	Displays the characteristics of a file system
<code>rmfs</code>	Removes a file system
<code>mount</code>	Makes a file system available for use

Four commands operate on virtual file systems types. The `/etc/vfs` file contains the information on the file system types that the following commands manipulate:

Item	Description
<code>chvfs</code>	Changes the characteristics of a file system type
<code>crvfs</code>	Adds a new file system type
<code>lsvfs</code>	Lists the characteristics of a file system type
<code>rmvfs</code>	Removes a file system type

Comparing file systems on different machines

When file systems that exist on different machines should be identical but you suspect one is damaged, you can compare the file systems.

The following procedure describes how to compare the attributes of a file system that resides on your current host (in this scenario, called *orig_host*) to the same file system on a remote host.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. Log in to the remote host as the root user.

For example:

```
tn juniper.mycompany.com

AIX Version 6.1
(C) Copyrights by IBM and by others 1982, 2002.
login: root
root's Password:
```

2. Using your favorite editor, edit the remote host's `.rhosts` file to add a stanza that allows the root user to execute secure remote commands. Use the following format for the new stanza:

```
orig_host root
```

The resulting `.rhosts` file might look similar to the following:

```
NIM.mycompany.com root
nim.mycompany.com root
host.othernetwork.com root
orig_host.mycompany.com root
```

3. Save your changes and exit the remote connection.
4. With root authority on *orig_host*, create another file using your favorite editor.

For this scenario, the new file is named compareFS. For example:

```
vi compareFS
```

5. Insert the following text in this file, where *FSname* is the name of the file system that you want to compare, and *remote_host* is the name of the host on which the comparison file system resides:

```
FSname -> remote_host  
install -v ;
```

Note: In the **install** command line of this file, there must be a space between the -v parameter and the semicolon (;).

For example:

```
/home/jane/* -> juniper.mycompany.com  
install -v ;
```

6. Save the file and exit the editor. The compareFS file is used as the *distfile* for the **rdist** command in the following step.
7. Type the following at the command prompt:

```
/usr/bin/rdist -f compareFS
```

Or, if you expect a significant amount of output from the comparison, send the output to a file name. For example:

```
/usr/bin/rdist -f compareFS > compareFS_output
```

The output lists any differences between the file systems.

Maintaining file systems

The simplest tasks you might need when maintaining file systems are grouped within this table.

Table 5. Maintaining File Systems Tasks		
Task	SMIT Fast Path	Command or File
Backup by name files or directories	smit backfile	backup ^{Note 1}
Create and back up a JFS2 snapshot image	smit backsnap	backsnap ^{Note 1}
List all file systems on a disk	smit lsmntdsk	
List file systems on a removable disk	smit lsmntdsk	
List mounted file systems	smit fs	
Mount a group of file systems ^{Note 5}	smit mountg	mount -t <i>GroupName</i>
Mount a JFS or JFS2 ^{Note 3}	smit mountfs	mount
Mount a JFS2 snapshot	smit mntsnap	mount -v jfs2 -o snapshot <i>Device MountPoint</i>
Remove a JFS or JFS2	smit rmfs	
Remove a JFS2 snapshot	smit rmsnap	snapshot -d <i>SnapshotDevice</i>
Revert a JFS2 file system to a point-in-time snapshot	smit rollbacksnap	rollback [-s] [-v] [-c] <i>snappedFS snapshotObject</i>

Table 5. Maintaining File Systems Tasks (continued)

Task	SMIT Fast Path	Command or File
Unmount a file system ^{Note 4}	smit umountfs	
Unmount a file system on a removable disk ^{Note 4}	smit umntdsk	
Unmount a group of file systems ^{Note 5}	smit umountg	umount -t GroupName
Manage Enhanced Journaled File Systems quotas	smit j2fsquotas	
Enable or disable quota management	smit j2enablequotas	
Stop/restart quota limits enforcement	smit j2enforcequotas	quotaon off -v
List quota usage	smit j2repquota	repquota -v
Recalculate current disk block and file usage statistics	smit j2quotacheck	quotacheck -v
Add a limits class	smit j2addlimit	j2edlimit -e
Change/show characteristics of a limits class	smit j2changelimit	
Make a limits class the default limits for a file system	smit j2defaultlimit	
Assign a user or group to a limits class	smit j2assignlimit	
List limits classes for a file system	smit j2listlimits	j2edlimit -l '-u'
Remove a limits class	smit j2removelimit	

Note:

1. For options, refer to the individual commands.
2. Do not change the names of system-critical file systems, which are / (root) on logical volume 4 (hd4), /usr on hd2, /var on hd9var, /tmp on hd3, and /b1v on hd5. If you use the hdn convention, start at hd10.
3. Check the file systems before mounting by using the procedure [“File system verification”](#) on page 89 or running the **fsck** command.
4. If an unmount fails, it might be because a user or process has an opened file in the file system being unmounted. The **fuser** command lets you find out which user or process might be causing the failure.
5. A file system group is a collection of file systems which have the same value for the **type=** identifier in the [/etc/filesystems](#) file.

Recovering one or more files from an online external JFS2 snapshot

You can replace a corrupted file if you have an accurate copy in an online external JFS2 snapshot.

Use the following procedure to recover one or more files from an online external JFS2 snapshot image.

For this example, assume that /home/aaa/myfile is a corrupted file in the /home file system.

1. Mount the snapshot with a command similar to the following:

```
mount -v jfs2 -o snapshot /dev/mysnaplv /tmp/mysnap
```

2. Change to the directory that contains the snapshot with a command similar to the following:

```
cd /tmp/mysnap
```

3. Copy the accurate file from the snapshot to overwrite the corrupted file with a command similar to the following:

```
cp aaa/myfile /home/aaa/myfile
```

The previous example copies only the file named `myfile`. If you want to copy all the files from the snapshot to the `aaa` directory, use a command similar to the following:

```
cp -R aaa /home/aaa
```

Recovering one or more files from an online internal JFS2 snapshot

You can replace a corrupted file if you have an accurate copy in an online internal JFS2 snapshot.

Use the following procedure to recover one or more files from an online internal JFS2 snapshot image.

For this example, assume that `/home/aaa/myfile` is a corrupted file in the `/home` file system.

1. Change to the directory that contains the snapshot with a command similar to the following:

```
cd /home/.snapshot/mysnap
```

2. Copy the accurate file from the snapshot to overwrite the corrupted file with a command similar to the following:

```
cp aaa/myfile /home/aaa/myfile
```

The previous example copies only the file named `myfile`. If you want to copy all of the files from the snapshot to the `aaa` directory, use a command similar to the following:

```
cp -R aaa /home/aaa
```

File systems on CD-ROM and DVD disks

CDs and DVDs are not automatically mounted, but this feature can be enabled.

To enable this feature, use the **`cdmount`** command to mount the CDRFS or UDFS file system, for example:

```
cdmount cd0
```

You can manually mount a read/write UDFS with the following command:

```
mount -V udfs DevName MtPt
```

Where *DevName* is the name of the DVD drive and *MtPt* is the mount point for the file system.

Using file systems on read/write optical media

CDRFS and JFS file systems can be used on read/write optical media.

A CD-ROM file system (CDRFS) can be stored on read/write optical media, provided that the optical media is write-protected, as well as on a CD-ROM. The following table tells you how to add, mount, or unmount a CDRFS on read/write optical media. You must specify the following information when mounting the file system:

Item	Description
Device name	Defines the name of device containing the media.

Item	Description
Mount point	Specifies the directory where the file system will be mounted.
Automatic mount	Specifies whether the file system will be mounted automatically at system restart.

CDRFS on Optical Media Tasks		
Task	SMIT Fast Path	Command or File
Adding a CDRFS ¹	smit crcdrfs	1. Add the file system: crfs -v cdrfs -p ro -dDeviceName -m MountPoint -A AutomaticMount 2. Mount the file system: mount MountPoint
Removing a CDRFS ²	1. Unmount the file system: smit umountfs 2. Remove the file system: smit rmcdrfs	1. Unmount the file system: umount FileSystem 2. Remove the file system: rmfs MountPoint

Note:

- Make sure the read/write optical media is write-protected.
- A CDRFS file system must be unmounted before the read/write optical media can be removed.

A JFS provides a read/write file system on optical media similar to those on a hard disk. You must have system authority to create or import a read/write file system on read/write optical media (that is, your login must belong to the system group) and you must have the following information:

Volume group name

Specifies the name of the volume group

Device name

Specifies the logical name of the read/write optical drive

Mount point

Specifies the directories where the file systems will be mounted

Automatic mount

Specifies whether the file system will be mounted automatically at system restart

Note:

- Any volume group created on read/write optical media must be self contained on that media. Volume groups cannot go beyond one read/write optical disk.
- When accessing a previously created journaled file system, the volume group name does not need to match the one used when the volume group was created.

JFS on Optical Media Tasks		
Task	SMIT Fast Path	Command or File

JFS on Optical Media Tasks		
Add a JFS	<ol style="list-style-type: none"> 1. Insert optical disk into drive. 2. Create a volume group (if necessary): smit mkvg 3. Create a journaled file system: smit crfs 	<ol style="list-style-type: none"> 1. Insert optical disk into drive. 2. Create a volume group (if necessary): mkvg -f -y VGName -d 1 DeviceName 3. Create a journaled file system: crfs -v jfs -g VGName -a size=SizeFileSystem -m MountPoint -A AutomaticMount -p rw 4. Mount the file system: mount MountPoint
Accessing previously created JFS ^{Note 1}	<ol style="list-style-type: none"> 1. Insert optical disk into drive. 2. Import the volume group: smit importvg 	<ol style="list-style-type: none"> 1. Insert optical disk into drive. 2. Import the volume group: importvg -y VGName DeviceName 3. Mount the file system: mount MountPoint
Removing a JFS ^{Note 2}	<ol style="list-style-type: none"> 1. Unmount the file system: smit umountfs 2. Remove the file system: smit rmjfs 	<ol style="list-style-type: none"> 1. Unmount the file system: umount FileSystem 2. Remove the file system: rmfs MountPoint

Note:

- This procedure is required whenever inserting media containing journaled file systems.
- Removing a journaled file system destroys all data contained in that file system and on the read/write optical media.

File system verification

Inconsistencies can occur in file systems when the system is stopped while file systems remained mounted or when a disk is damaged. In such circumstances, it is important to verify file systems before mounting them.

Also verify your file systems in the following circumstances:

- After a malfunction; for example, if a user cannot change directories to a directory that has that user's permissions (uid)
- Before backing up file systems to prevent errors and possible restoration problems
- At installation or system boot to make sure that there are no operating system file errors

Checking a user-defined file system

To check a user-defined file system, perform the following steps.

1. Unmount the user-defined file system being checked.
2. Ensure you have write permission on files in the file system. Otherwise, the **fsck** cannot repair damaged files even if you answer Yes to repair prompts.
3. Use the **smit fsck** fast path to access the **Verify a File System** menu.
4. Do one of the following:
 - Specify the name of an individual file system to check in the **NAME of file system** field, or

- Select a general file system type to check, such as a journaled file system (JFS) in the **TYPE of file system** field.
5. If you want to limit your check to the most likely candidates, specify Yes in the **FAST check?** field.
The fast-check option checks only those file systems that are likely to have inconsistencies such as the file systems that were mounted when the system stopped at some point in the past.
 6. Specify the name of a temporary file on a file system not being checked in the **SCRATCH file** field.
 7. Start the file system check.

Checking root and /usr file systems

To run the **fsck** command on / or /usr file system, you must shut down the system and reboot it from removable media because the / (root) and /usr file systems cannot be unmounted from a running system.

The following procedure describes how to run **fsck** on the / and /usr file systems from the maintenance shell.

1. Shut down your system
(Root access required)
2. Boot from your installation media.
3. From the **Welcome** menu, choose the **Maintenance** option.
4. From the **Maintenance** menu, choose the option to access a volume group.
5. Choose the rootvg volume group. A list of logical volumes that belong to the volume group you selected is displayed.
6. Choose **2** to access the volume group and to start a shell before mounting file systems.
In the following steps, you will run the **fsck** command using the appropriate options and file system device names. The **fsck** command checks the file system consistency and interactively repairs the file system. The / (root) file system device is /dev/hd4 and the /usr file system device is /dev/hd2.
7. To check / file system, type the following:

```
$ fsck -y /dev/hd4
```

The **-y** flag is recommended for less experienced users (see the **fsck** command).

8. To check the /usr file system, type the following:

```
$ fsck -y /dev/hd2
```

9. To check other file systems in the rootvg, type the **fsck** command with the appropriate device names. The device for /tmp is /dev/hd3, and the device for /var is /dev/hd9var.
10. When you have completed checking the file systems, reboot the system.

Reducing the size of a file system in your root volume group

The simplest way to reduce *all* file systems to their minimum size is to set the **SHRINK** option to **yes** when restoring the base operating system from backup.

The simplest way to reduce *all* file systems to their minimum size is to set the **SHRINK** option to **yes** when restoring the base operating system from backup. The **SHRINK** option and the following scenario cannot be used in tandem. If you set the **SHRINK** option to **yes** after doing the following procedure, the installation overrides your changes to the /image.data file.

This scenario leads you through a manual process to reduce the size of a selected rootvg file system. You will identify a file system that is not using all of its allocated disk space and then reallocate based on the amount of space the file system actually uses, thus freeing more space for the root volume group's use. As part of this procedure, you will back up your volume groups and reinstall the operating system, using the revised allocations.



Attention: This procedure requires shutting down and reinstalling the base operating system. Whenever you reinstall any operating system, schedule your downtime when it least impacts your workload to protect yourself from a possible loss of data or functionality. Before reinstalling the operating system, ensure you have reliable backups of your data and any customized applications or volume groups.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. Create a separate backup of all file systems that are *not* contained in the rootvg. The separate backup helps ensure the integrity of all your file systems.
2. With root authority, check which file systems in your root volume group are not using their allocated disk space by typing the following command:

```
df -k
```

The **-k** flag displays the file-system sizes in kilobytes. Your result will look similar to the following:

Filesystem	1024-blocks	Free	%Used	Iused	%Iused	Mounted on
/dev/hd4	196608	4976	98%	1944	2%	/
/dev/hd2	1769472	623988	65%	36984	9%	/usr
/dev/hd9var	163840	65116	61%	676	2%	/var
/dev/hd3	65536	63024	4%	115	1%	/tmp
/dev/hd1	49152	8536	83%	832	7%	/home
/proc	-	-	-	-	-	/proc
/dev/hd10opt	32768	26340	20%	293	4%	/opt

Looking at these results, you notice a large number of free blocks and a fairly low percentage of use associated for the file system that is mounted on `/usr`. You decide you can release a significant number of blocks by reducing the number of partitions allocated to the `/usr` file system.

3. Check the contents of the `/etc/filesystems` file to ensure that all file systems in the rootvg are mounted. If not, they will not be included in the reinstalled system.
4. Create an `/image.data` file, which lists all the active file systems in the rootvg that are included in the installation procedure, by typing the following command:

```
mkszfile
```

5. Open the `/image.data` file in your favorite editor.
6. Search for the `usr` text string to locate the `lv_data` stanza that pertains to the `/usr` file system. Use numbers from this stanza as a base to determine how much you can reduce the `/usr` file system's number of logical partitions. The default size of each additional logical partition is defined in the `PP_SIZE` entry of the `/image.data` file. Your `/image.data` file would look similar to the following:

```
lv_data:
  VOLUME_GROUP= rootvg
  LV_SOURCE_DISK_LIST= hdisk0
  LV_IDENTIFIER= 00042345d300bf15.5
  LOGICAL_VOLUME= hd2
  VG_STAT= active/complete
  TYPE= jfs
  MAX_LPS= 32512
  COPIES= 1
  LPS= 108
  STALE_PPS= 0
  INTER_POLICY= minimum
  INTRA_POLICY= center
  MOUNT_POINT= /usr
  MIRROR_WRITE_CONSISTENCY= on/ACTIVE
  LV_SEPARATE_PV= yes
  PERMISSION= read/write
  LV_STATE= opened/syncd
  WRITE_VERIFY= off
  PP_SIZE= 16
  SCHED_POLICY= parallel
  PP= 108
  BB_POLICY= relocatable
  RELOCATABLE= yes
```

```
UPPER_BOUND= 32
LABEL= /usr
MAPFILE=
LV_MIN_LPS= 70
STRIPE_WIDTH=
STRIP_SIZE=
```

The number of logical partitions devoted to this logical volume is 108 (LPs=108).

7. Determine the number of logical partitions needed by the existing data in the /usr file system by using your result from step 2.

You can display the existing file sizes specifically for the /usr file system by using the following command:

```
df -k /usr
```

The result repeats the numbers (in kilobytes) you received for the /usr file system in step 2. For example:

Filesystem	1024-blocks	Free	%Used	Iused	%Iused	Mounted on
/dev/hd2	1769472	623988	65%	36984	9%	/usr

- a) Subtract the amount of free space from the total number of 1024-blocks allocated:

```
1769472 - 623988 = 1145484
```

- b) Add an estimate of the space you might need to accommodate any future growth expected in this file system.

For this example, add 200000 to the result.

```
1145484 + 200000 = 1345484
```

- c) Divide the result by the logical-partition size in bytes (16*1024) to determine the minimum number of logical partitions you need.

```
1345484 / 16384 = 82.121826171875
```

Use this result, rounded upward, to redefine the number of logical partitions needed (LPs=83).

8. In your image.data file, change the LPs field from 108 to 83.

9. Find the fs_data stanza that pertains to the /usr file system.

Your fs_data stanza will look similar to the following:

```
fs_data:
  FS_NAME= /usr
  FS_SIZE= 3538944
  FS_MIN_SIZE= 2290968
  FS_LV= /dev/hd2
  FS_FS= 4096
  FS_NBPI= 4096
  FS_COMPRESS= no
  FS_BF= false
  FS_AGSIZE= 8
```

10. Calculate the file-system size (FS_SIZE) by multiplying the physical partition size (PP_SIZE) by 2 (the number of 512-byte blocks used by the physical partitions) by the number of logical partitions (LPs).

Given the values used in this example, the calculation is:

```
PP_SIZE * 512 blocks * LPs = FS_SIZE
16384 * 2 * 83 = 2719744
```

11. In your image.data file, change the FS_SIZE field from 3538944 to 2719744.
12. Calculate the minimum file-system size (FS_MIN_SIZE) based on the actual size of the current data used by the /usr file system, as follows:
 - a) Calculate the minimum number of partitions needed.

Given the values used in this example, the calculation is:

$\text{size_in_use (see step 7a) / PP_SIZE = partitions}$
 $1145484 / 16384 = 69.914794921875$

- b) Calculate the minimum size required by that number of partitions.
Rounding the previous calculation results upward to 70, the calculation is:

```
PP_SIZE * 512 blocks * partitions = FS_MIN_SIZE  
16384 * 2 * 70 = 2293760
```

13. In your `image.data` file, change the `FS_MIN_SIZE` field from 2290968 to 2293760.
14. Save your edits and exit the editor.
15. Unmount all file systems that are not in the rootvg volume group.
16. If you have any user-defined volume groups, type the following commands to vary off and export them:

```
varyoffvg VGName  
exportvg VGName
```

17. With a tape in the tape drive, type the following command to initiate a complete system backup:

```
mksysb /dev/rmt0
```

This type of backup includes the file-system size information you specified in the `/image.data` file, which will be used later to reinstall your system with the new file-system sizes.

Note: To initiate this backup, you must run the **mksysb** command from the command line. If you use a system management tool, such as SMIT, the backup creates a new `image.data` file, overwriting the changes you have made.

18. Use this backup to reinstall the operating system using the **Install With Current System Settings** option.

During the installation, check that the following options are set appropriately:

- **Use Maps** must be set to **no**
- **Shrink the File Systems** must be set to **no**

If you need more information about the installation procedure, see the [Installing system backups](#).

19. After the operating system is installed, reboot the system in Normal mode.

At this point, the `/usr` file system is resized, but your user-defined file systems are not available.

20. Mount all file systems by typing the following command:

```
mount all
```

If you receive `Device Busy` messages about file systems that are already mounted, you can ignore these messages.

At this point, your `/usr` file system is resized, your root volume group has more free space, and your file systems are usable.

Related concepts

[Logical volume storage](#)

Logical volumes are groups of information located on physical volumes.

Related information

[Creating a Root Volume Group Backup to Tape or File](#)

[/image.data file description](#)

[mkszfile command](#)

[mksysb command](#)

Troubleshooting file systems

Use these troubleshooting methods to tackle some of the basic problems that may occur to your file systems. If the troubleshooting information does not address your problem, contact your service representative.

Fixing a user-defined file system overflow

Use this procedure to fix an overflowing user-defined file system.

1. Remove old backup files and core files. The following example removes all *.bak, *.bak, a.out, core, *, or ed.hup files.

```
find / \( -name "*.bak" -o -name core -o -name a.out -o \  
-name "...*" -o -name "*.bak" -o -name ed.hup \) \  
-atime +1 -mtime +1 -type f -print | xargs -e rm -f
```

2. To prevent files from regularly overflowing the disk, run the **skulker** command as part of the **cron** process and remove files that are unnecessary or temporary.

The **skulker** command purges files in /tmp directory, files older than a specified age, a.out files, core files, and ed.hup files. It is run daily as part of an accounting procedure run by the **cron** command during off-peak periods (assuming you have turned on accounting).

The **cron** daemon runs shell commands at specified dates and times. Regularly scheduled commands such as **skulker** can be specified according to instructions contained in the crontab files. Submit crontab files with the **crontab** command. To edit a crontab file, you must have root user authority.

Fixing a damaged file system

File systems can get corrupted when the i-node or superblock information for the directory structure of the file system gets corrupted.

This corruption can be caused by a hardware-related ailment or by a program that gets corrupted that accesses the i-node or superblock information directly. (Programs written in assembler and C can bypass the operating system and write directly to the hardware.) One symptom of a corrupt file system is that the system cannot locate, read, or write data located in the particular file system.

To fix a damaged file system, you must diagnose the problem and then repair it. The **fsck** command performs low-level diagnosis and repairs.

The following is the procedure for fixing a damaged file system:

1. With root authority, unmount the damaged file system using one of the following SMIT fast paths: **smit unmountfs** (for a file system on a fixed disk drive) or **smit unmntdsk** (for a file system on a removeable disk).
2. Assess file system damage by running the **fsck** command. In the following example, the **fsck** command checks the unmounted file system located on the /dev/myfilelv device:

```
fsck /dev/myfilelv
```

The **fsck** command checks and interactively repairs inconsistent file systems. Normally, the file system is consistent, and the **fsck** command merely reports on the number of files, used blocks, and free blocks in the file system. If the file system is inconsistent, the **fsck** command displays information about the inconsistencies found and prompts you for permission to repair them. The **fsck** command is conservative in its repair efforts and tries to avoid actions that might result in the loss of valid data. In certain cases, however, the **fsck** command recommends the destruction of a damaged file.

3. If the file system cannot be repaired, restore it from backup.



Attention: Restoring a file system from a backup destroys and replaces any file system previously stored on the disk.

To restore the file system from backup, use the SMIT fastpath **smit restfilesys** or the series of commands shown in the following example:

```
mkfs /dev/myfilelv
mount /dev/myfilelv /myfilesys
cd /myfilesys
restore -r
```

In this example, the **mkfs** command makes a new file system on the device named `/dev/myfilelv` and initializes the volume label, file system label, and startup block. The **mount** command establishes `/dev/myfilelv` as the mountpoint for **myfilesys** and the **restore** command extracts the file system from the backup.

If your backup was made using incremental file system backups, you must restore the backups in increasing backup-level order (for example, 0, 1, 2). When using **smit restfilesys** to restore an entire file system, enter the target directory, restore device (other than `/dev/rfd0`), and number of blocks to read in a single input operation.

Fixing a corrupted magic number in the file system superblock

If the superblock of a file system is damaged, the file system cannot be accessed. You can fix a corrupted magic number in the file system superblock.

Most damage to the superblock cannot be repaired. The following procedure describes how to repair a superblock in a JFS file system when the problem is caused by a corrupted magic number. If the primary superblock is corrupted in a JFS2 file system, use the **fsck** command to automatically copy the secondary superblock and repair the primary superblock.

In the following scenario, assume `/home/myfs` is a JFS file system on the physical volume `/dev/lv02`.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

1. Unmount the `/home/myfs` file system, which you suspect might be damaged, using the following command:

```
umount /home/myfs
```

2. To confirm damage to the file system, run the **fsck** command against the file system. For example:

```
fsck -p /dev/lv02
```

If the problem is damage to the superblock, the **fsck** command returns one of the following messages:

```
fsck: Not an AIXV5 file system
```

OR

```
Not a recognized filesystem type
```

3. With root authority, use the **od** command to display the superblock for the file system, as shown in the following example:

```
od -x -N 64 /dev/lv02 +0x1000
```

Where the `-x` flag displays output in hexadecimal format and the `-N` flag instructs the system to format no more than 64 input bytes from the offset parameter (+), which specifies the point in the file where the file output begins. The following is an example output:

```
0001000 1234 0234 0000 0000 0000 4000 0000 000a
0001010 0001 8000 1000 0000 2f6c 7633 0000 6c76
0001020 3300 0000 000a 0003 0100 0000 2f28 0383
0001030 0000 0001 0000 0200 0000 2000 0000 0000
0001040
```

In the preceding output, note the corrupted magic value at 0x1000 (1234 0234). If all defaults were taken when the file system was created, the magic number should be 0x43218765. If any defaults were overridden, the magic number should be 0x65872143.

4. Use the **od** command to check the secondary superblock for a correct magic number. An example command and its output follows:

```
$ od -x -N 64 /dev/lv02 +0x1f000
001f000 6587 2143 0000 0000 0000 4000 0000 000a
001f010 0001 8000 1000 0000 2f6c 7633 0000 6c76
001f020 3300 0000 000a 0003 0100 0000 2f28 0383
001f030 0000 0001 0000 0200 0000 2000 0000 0000
001f040
```

Note the correct magic value at 0x1f000.

5. Copy the secondary superblock to the primary superblock. An example command and output follows:

```
$ dd count=1 bs=4k skip=31 seek=1 if=/dev/lv02 of=/dev/lv02
dd: 1+0 records in.
dd: 1+0 records out.
```

6. Use the **fsck** command to clean up inconsistent files caused by using the secondary superblock. For example:

```
fsck /dev/lv02 2>&1 | tee /tmp/fsck.errs
```

Related information

[fsck command](#)

[od command](#)

Disk overflows

A disk overflow occurs when too many files fill up the allotted space. This can be caused by a runaway process that creates many unnecessary files.

You can use the following procedures to correct the problem:

Note: You must have root user authority to remove processes other than your own.

Identifying problem processes

Use this procedure to isolate problem processes.

1. To check the process status and identify processes that might be causing the problem, type:

```
ps -ef | pg
```

The **ps** command shows the process status. The **-e** flag writes information about all processes (except kernel processes), and the **-f** flag generates a full listing of processes including what the command name and parameters were when the process was created. The **pg** command limits output to a single page at a time, so information does not scroll too quickly off the screen.

Check for system or user processes that are using excessive amounts of a system resource, such as CPU time. System processes such as **sendmail**, **routed**, and **lpd** seem to be the system processes most prone to becoming runaways.

2. To check for user processes that use more CPU than expected, type:

```
ps -u
```

3. Note the process ID (PID) of each problem process.

Terminating a process

You can terminate problem processes.

Use the following procedure to terminate a problem process:

1. Terminate the process that is causing the problem by typing:

```
kill -9 PID
```

Where *PID* is the ID of the problem process.

2. Remove the files the process has been making by typing:

```
rm file1 file2 file3
```

Where *file1 file2 file3* represents names of process-related files.

Reclamation of file space without terminating a process

To reclaim the blocks allocated to an active file without terminating the process, redirect the output of another command to the file. The data redirection truncates the file and reclaims the blocks of memory.

When the active file is removed from the file system, the blocks allocated to the file remain allocated until the last open reference is removed, either as a result of the process closing the file or because of the termination of the processes that have the file open. If a runaway process is writing to a file and the file is removed, the blocks allocated to the file are not freed until the process terminates.

For example:

```
$ ls -l
total 1248
-rwxrwxr-x    1 web    staff    1274770 Jul 20 11:19 datafile
$ date > datafile
$ ls -l
total 4
-rwxrwxr-x    1 web    staff          29 Jul 20 11:20 datafile
```

The output of the **date** command replaced the previous contents of the datafile file. The blocks reported for the truncated file reflect the size difference from 1248 to 4. If the runaway process continues to append information to this newly truncated file, the next **ls** command produces the following results:

```
$ ls -l
total 8
-rwxrwxr-x    1 web    staff    1278866 Jul 20 11:21 datafile
```

The size of the datafile file reflects the append done by the runaway process, but the number of blocks allocated is small. The datafile file now has a hole in it. File holes are regions of the file that do not have disk blocks allocated to them.

/ (root) overflow

Check the following when the root file system (/) has become full.

- Use the following command to read the contents of the `/etc/security/failedlogin` file:

```
who /etc/security/failedlogin
```

The condition of TTYs recreating too rapidly can create failed login entries. To clear the file after reading or saving the output, execute the following command:

```
cp /dev/null /etc/security/failedlogin
```

- Check the `/dev` directory for a device name that is typed incorrectly. If a device name is typed incorrectly, such as `rmt0` instead of `rmt0`, a file will be created in `/dev` called `rmt0`. The command will normally proceed until the entire root file system is filled before failing. `/dev` is part of the root (/)

file system. Look for entries that are not devices (that do not have a major or minor number). To check for this situation, use the following command:

```
cd /dev
ls -l | pg
```

In the same location that would indicate a file size for an ordinary file, a device file has two numbers separated by a comma. For example:

```
crw-rw-rw-  1 root    system   12,0 Oct 25 10:19 rmt0
```

If the file name or size location indicates an invalid device, as shown in the following example, remove the associated file:

```
crw-rw-rw-  1 root    system 9375473 Oct 25 10:19 rmt0
```

Note:

- Do not remove valid device names in the /dev directory. One indicator of an invalid device is an associated file size that is larger than 500 bytes.
- If system auditing is running, the default /audit directory can rapidly fill up and require attention.
- Check for very large files that might be removed using the **find** command. For example, to find all files in the root (/) directory larger than 1 MB, use the following command:

```
find / -xdev -size +2048 -ls |sort -r -n +6
```

This command finds all files greater than 1 MB and sorts them in reverse order with the largest files first. Other flags for the find command, such as **-newer**, might be useful in this search. For detailed information, see the command description for the **find** command.

Note: When checking the root directory, major and minor numbers for devices in the /dev directory will be interspersed with real files and file sizes. Major and minor numbers, which are separated by a comma, can be ignored.

Before removing any files, use the following command to ensure a file is not currently in use by a user process:

```
fuser filename
```

Where *filename* is the name of the suspect large file. If a file is open at the time of removal, it is only removed from the directory listing. The blocks allocated to that file are not freed until the process holding the file open is killed.

Resolving overflows in the /var file system

Check the following when the /var file system has become full.

- You can use the find command to look for large files in the /var directory. For example:

```
find /var -xdev -size +2048 -ls | sort -r +6
```

For detailed information, see the command description for the **find** command.

- Check for obsolete or leftover files in /var/tmp.
- Check the size of the /var/adm/wtmp file, which logs all logins, rlogins and telnet sessions. The log will grow indefinitely unless system accounting is running. System accounting clears it out nightly. The /var/adm/wtmp file can be cleared out or edited to remove old and unwanted information. To clear it, use the following command:

```
cp /dev/null /var/adm/wtmp
```

To edit the `/var/adm/wtmp` file, first copy the file temporarily with the following command:

```
/usr/sbin/acct/fwtmp < /var/adm/wtmp > /tmp/out
```

Edit the `/tmp/out` file to remove unwanted entries then replace the original file with the following command:

```
/usr/sbin/acct/fwtmp -ic < /tmp/out > /var/adm/wtmp
```

- Clear the error log in the `/var/adm/ras` directory using the following procedure. The error log is never cleared unless it is manually cleared.

Note: Never use the `cp /dev/null` command to clear the error log. A zero-length `errlog` file disables the error logging functions of the operating system and must be replaced from a backup.

1. Stop the error daemon using the following command:

```
/usr/lib/errstop
```

2. Remove or move to a different filesystem the error log file by using one of the following commands:

```
rm /var/adm/ras/errlog
```

or

```
mv /var/adm/ras/errlog filename
```

Where *filename* is the name of the moved `errlog` file.

Note: The historical error data is deleted if you remove the error log file.

3. Restart the error daemon using the following command:

```
/usr/lib/errdaemon
```

Note: Consider limiting the `errlog` by running the following entries in **cron**:

```
0 11 * * * /usr/bin/errclear -d S,0 30
0 12 * * * /usr/bin/errclear -d H 90
```

- Check whether the `trcfile` file in this directory is large. If it is large and a trace is not currently being run, you can remove the file using the following command:

```
rm /var/adm/ras/trcfile
```

- If your dump device is set to `hd6` (which is the default), there might be a number of `vmcore*` files in the `/var/adm/ras` directory. If their file dates are old or you do not want to retain them, you can remove them with the `rm` command.
- Check the `/var/spool` directory, which contains the queueing subsystem files. Clear the queueing subsystem using the following commands:

```
stopsrc -s qdaemon
rm /var/spool/lpd/qdir/*
rm /var/spool/lpd/stat/*
rm /var/spool/qdaemon/*
startsrc -s qdaemon
```

- Check the `/var/adm/acct` directory, which contains accounting records. If accounting is running, this directory may contain several large files.
- Check the `/var/preserve` directory for terminated `vi` sessions. Generally, it is safe to remove these files. If a user wants to recover a session, you can use the `vi -r` command to list all recoverable sessions. To recover a specific session, use `vi -r filename`.
- Modify the `/var/adm/sulog` file, which records the number of attempted uses of the `su` command and whether each was successful. This is a flat file and can be viewed and modified with a favorite editor. If it is removed, it will be recreated by the next attempted `su` command. Modify the `/var/tmp/`

snmpd.log, which records events from the **snmpd** daemon. If the file is removed it will be recreated by the **snmpd** daemon.

Note: The size of the /var/tmp/snmpd.log file can be limited so that it does not grow indefinitely. Edit the /etc/snmpd.conf file to change the number (in bytes) in the appropriate section for size.

Fix other file systems and general search techniques

Use the **find** command with the **-size** flag to locate large files or, if the file system recently overflowed, use the **-newer** flag to find recently modified files.

To produce a file for the **-newer** flag to find against, use the following touch command:

```
touch mmddhhmm filename
```

Where *mm* is the month, *dd* is the date, *hh* is the hour in 24-hour format, *mm* is the minute, and *filename* is the name of the file you are creating with the **touch** command.

After you have created the touched file, you can use the following command to find newer large files:

```
find /filesystem_name -xdev -newer touch_filename -ls
```

You can also use the **find** command to locate files that have been changed in the last 24 hours, as shown in the following example:

```
find /filesystem_name -xdev -mtime 0 -ls
```

Mounting

Mounting makes file systems, files, directories, devices, and special files available for use at a particular location. It is the only way a file system is made accessible.

The **mount** command instructs the operating system to attach a file system at a specified directory.

You can mount a file or directory if you have access to the file or directory being mounted and write permission for the mount point. Members of the system group can also perform device mounts (in which devices or file systems are mounted over directories) and the mounts described in the /etc/filesystems file. A user operating with root user authority can mount a file system arbitrarily by naming both the device and the directory on the command line. The /etc/filesystems file is used to define mounts to be automatic at system initialization. The **mount** command is used to mount after system startup.

Mount points

A *mount point* is a directory or file at which a new file system, directory, or file is made accessible. To mount a file system or a directory, the mount point must be a directory; and to mount a file, the mount point must be a file.

Typically, a file system, directory, or file is mounted over an empty mount point, but that is not required. If the file or directory that serves as the mount point contains any data, that data is not accessible while it is mounted over by another file or directory. In effect, the mounted file or directory covers what was previously in that directory. The original directory or file that has been mounted over is accessible again once the mount over it is undone.

When a file system is mounted over a directory, the permissions of the root directory of the mounted file system take precedence over the permissions of the mount point. The one exception involves the .. (dot dot) parent directory entry in the mounted-over directory. In order for the operating system to access the new file system, the mount point parent directory information must be available.

For example, if the current working directory is /home/frank, the command **cd ..** changes the working directory to /home. If /home/frank directory is the root of a mounted file system, the operating system must find the parent directory information in the /home/frank directory in order for the **cd ..** command to succeed.

For any command that requires parent directory information in order to succeed, users must have search permission in the mounted-over directory. Failure of the mounted-over directory to grant search permission can have unpredictable results, especially since the mounted-over directory permissions are not visible. A common problem is failure of the **pwd** command. Without search permission in the mounted-over directory, the **pwd** command returns this message:

```
pwd: Permission denied
```

This problem can be avoided by always setting the permissions of the mounted-over directory to at least 111.

Mounting file systems, directories, and files

There are two types of mounts, a remote mount and a local mount. *Remote mounts* are done on a remote system on which data is transmitted over a telecommunication line. Remote file systems, such as Network File System (NFS), require that the files be exported before they can be mounted. *Local mounts* are mounts done on your local system.

Each file system is associated with a different device (logical volume). Before you can use a file system, it must be connected to the existing directory structure (either the root file system or to another file system that is already connected). The **mount** command makes this connection.

The same file system, directory, or file can be accessed by multiple paths. For example, if you have one database and several users using this database, it can be useful to have several mounts of the same database. Each mount should have its own name and password for tracking and job-separating purposes. This is accomplished by mounting the same file system on different mount points. For example, you can mount from `/home/server/database` to the mount point specified as `/home/user1`, `/home/user2`, and `/home/user3`:

```
/home/server/database    /home/user1
/home/server/database    /home/user2
/home/server/database    /home/user3
```

A file system, directory, or file can be made available to various users through the use of symbolic links. Symbolic links are created with the **ln -s** command. Linking multiple users to a central file ensures that all changes to the file are reflected each time a user accesses the file.

Automatic mount control

Mounts can be set to occur automatically during system initialization.

There are two types of automatic mounts. The first type consists of those mounts that are required to boot and run the system. These file systems are explicitly mounted by the boot process. The stanzas of such file systems in the `/etc/filesystems` file have `mount = automatic`. The second type of automatic mount is user-controlled. These file systems are mounted by the `/etc/rc` script when it issues the **mount all** command. The stanzas of user-controlled automatic mounts have `mount = true` in `/etc/filesystems`.

The `/etc/filesystems` file controls automatic mounts; they are done hierarchically, one mount point at a time. They can also be placed in a specific order that can be changed and rearranged. For more information about the `/etc/filesystems` file, see [/etc/filesystems](#).

The `/etc/filesystems` file is organized into stanzas, one for each mount. A stanza describes the attributes of the corresponding file system and how it is mounted. The system mounts file systems in the order they appear in the `/etc/filesystems` file. The following is an example of stanzas within the `/etc/filesystems` file:

```
/:
dev=/dev/hd4
vol="root"
mount=automatic
check=false
free=true
vfs=jfs
```

```

log=/dev/hd8
type=bootfs

/home:
dev=/dev/hd1
vfs=jfs
log=/dev/hd8
mount=true
check=true
vol="/home"
free=false

/usr:
/dev=/dev/hd2
vfs=jfs
log=/dev/hd8
mount=automatic
check=false
type=bootfs
vol="/usr"
free=false

```

You can edit the `/etc/filesystems` file to control the order in which mounts occur. If a mount is unsuccessful, any of the following mounts defined in the `/etc/filesystems` file continue to mount. For example, if the mount of the `/home` file system is unsuccessful, the mount for the `/usr` file system continues and be mounted. Mounts can be unsuccessful for reasons such as typographical errors, dependency, or a system problem.

Mount security for diskless workstations

Diskless workstations must have the ability to create and access device-special files on remote machines to have their `/dev` directories mounted from a server. Because servers cannot distinguish device-special files intended for a client from those intended for the server, a user on the server might be able to access the physical devices of the server using the special files of the client device.

For example, the ownership for a **tty** is automatically set to the user using the **tty**. If the user IDs are not the same on both the client and server, a nonprivileged user on the server can access a **tty** that is being used by a different user on the server.

A user who is privileged on a client can create device-special files to match physical devices on the server and have them not require privilege for access. The user can then use an unprivileged account on the server to access the normally protected devices using the new device-special files.

A similar security problem involves the use of **setuid** and **setgid** programs on the client and server. Diskless clients must be able to create and run **setuid** and **setgid** programs on the server for normal operation. Again, the server cannot distinguish between those programs intended for the server and those intended for the client.

In addition, the user IDs and group IDs might not match between the server and client, so users on the server might be able to run programs with capabilities that were not intended for them.

The problem exists because the **setuid** and **setgid** programs and device-special files should only be usable on the machine that created them.

The solution is to use security options to the **mount** command that restrict the ability to use these objects. These options can also be used in stanzas in the `/etc/filesystems` file.

The **nosuid** option in the **mount** command prevents the execution of **setuid** and **setgid** programs that are accessed via the resulting mounted file system. This option is used for any file system that is being mounted on a particular host for use only by a different host (for example, exported for diskless clients).

The **nodev** option in the **mount** command prevents the opening of devices using device special files that are accessed via the resulting mounted file system. This option is also used for any file system that is being mounted for use only by a different host (for example, exported for diskless clients).

In general, users on a server do not have any access to the `/export` directory.

Exporting the /export/root Directory

The /export/root directory must be exported with read/write permissions, and the root user on the server must have access. However, you might want to mount this directory with the following options of the **mount** command:

Item	Description
------	-------------

nosuid	Prevents a user on the server from running the setuid programs of the client
---------------	---

nodev	Prevents a user from accessing the server devices using a device-special file of the client.
--------------	--

An alternative to mounting the /export/root directory with these options is to avoid giving users running on the server any access to the /export/root directory.

Exporting the /export/exec Directory

The /export/exec directory is exported with read-only permissions and must provide root access. However, you might want to mount this directory with the following options of the **mount** command:

Item	Description
------	-------------

nosuid	Prevents a user on the server from running the setuid programs of the client. If you are exporting the server /usr directory, you cannot use the nosuid option.
---------------	---

nodev	Prevents a user from accessing the server devices using a device-special file of the client.
--------------	--

Exporting the /export/share Directory

The /export/share directory is exported with read-only permissions and must provide root access. Because this directory generally contains only data (no executables or devices), you do not need to use the mount security options.

Exporting the /export/home Directory

There are several ways to mount a user /home directory:

- You can mount the /export/home/*Clienthostname* directory over the client /home directory. In this case, the client has read/write permissions and the root user has access. To ensure system security, mount the /export/home directory with the following options to the **mount** command:

Item	Description
------	-------------

nosuid	Prevents a user on the server from running the setuid programs of the client.
---------------	--

nodev	Prevents a user from accessing the server devices using a device-special file of the client.
--------------	--

- You can mount the /home directory on the server over the /home directory of the client. In this case, the /home directory is exported with read/write permissions and without root access. To ensure system security, mount the /home directory on both the server and client with the **nosuid** and **nodev** options of the **mount** command.
- Alternatively, you can mount on the client each /home/*UserName* directory on the server over the /home/*Username* directory on the client so users can log in to different machines and still have access to their home directories. In this case, the /home/*Username* directories on the server and clients are both mounted with the **nosuid** and **nodev** options of the **mount** command.

Exporting the /export/swap Directory

Export the /export/swap/*Clienthostname* file with read/write permissions and root access. No security measures are necessary. Users on the server do not have any access to the /export/swap/*Clienthostname* files.

Diskless mounts

Although the file system of a diskless workstation is mounted from a server `/exports` directory, to the diskless machine, the file system looks just like the file system on a standalone machine.

The following shows the relationship between server exports, and the diskless workstation mount points:

Server Exports	Diskless Imports
<code>/export/root/HostName</code>	<code>/ (root)</code>
<code>/export/exec/SPOTName</code>	<code>/usr</code>
<code>/export/home/HostName</code>	<code>/home</code>
<code>/export/share</code>	<code>/usr/share</code>
<code>/export/dump</code>	Used by diskless client as dump space
<code>/export/swap</code>	Used by diskless clients as remote paging space

For more information about the `/export` directory, see [“/export directory” on page 81](#).

In general, users on a server do not have any access to the `/export` directory.

Exporting the `/export/root` Directory

The `/export/root` directory must be exported with read/write permissions, and the root user on the server must have access. However, you might want to mount this directory with the following options of the **mount** command:

Item	Description
nosuid	Prevents a user on the server from running the setuid programs of the client
nodev	Prevents a user from accessing the server devices using a device-special file of the client.

An alternative to mounting the `/export/root` directory with these options is to avoid giving users running on the server any access to the `/export/root` directory.

Exporting the `/export/exec` Directory

The `/export/exec` directory is exported with read-only permissions and must provide root access. However, you might want to mount this directory with the following options of the **mount** command:

Item	Description
nosuid	Prevents a user on the server from running the setuid programs of the client. If you are exporting the server <code>/usr</code> directory, you cannot use the nosuid option.
nodev	Prevents a user from accessing the server devices using a device-special file of the client.

Exporting the `/export/share` Directory

The `/export/share` directory is exported with read-only permissions and must provide root access. Because this directory generally contains only data (no executables or devices), you do not need to use the mount security options.

Exporting the `/export/home` Directory

There are several ways to mount a user `/home` directory:

- You can mount the `/export/home/Clienthostname` directory over the client `/home` directory. In this case, the client has read/write permissions and the root user has access. To ensure system security, mount the `/export/home` directory with the following options to the **mount** command:

Item	Description
------	-------------

- | | |
|---------------|--|
| nosuid | Prevents a user on the server from running the setuid programs of the client. |
| nodev | Prevents a user from accessing the server devices using a device-special file of the client. |

- You can mount the `/home` directory on the server over the `/home` directory of the client. In this case, the `/home` directory is exported with read/write permissions and without root access. To ensure system security, mount the `/home` directory on both the server and client with the **nosuid** and **nodev** options of the **mount** command.
- Alternatively, you can mount on the client each `/home/UserName` directory on the server over the `/home/Username` directory on the client so users can log in to different machines and still have access to their home directories. In this case, the `/home/Username` directories on the server and clients are both mounted with the **nosuid** and **nodev** options of the **mount** command.

Exporting the `/export/dump` Directory

Export the `/export/dump/Clienthostname` directory with read/write permissions and root access. Users on the server do not have any access to the `/export/dump/Clienthostname` files.

Exporting the `/export/swap` Directory

Export the `/export/swap/Clienthostname` file with read/write permissions and root access. No security measures are necessary. Users on the server do not have any access to the `/export/swap/Clienthostname` files.

File system types

AIX supports multiple file system types.

These include the following:

Journalized File System (JFS) or Enhanced Journalized File System (JFS2)

Supports the entire set of file system semantics. These file systems use database journaling techniques to maintain structural consistency. This prevents damage to the file system when the system is halted abnormally.

Each JFS or JFS2 resides on a separate logical volume. The operating system mounts the file system during initialization. This multiple file system configuration is useful for system management functions such as backup, restore, and repair, because it isolates a part of the file tree so that you can work on it.

JFS is the basic file system type that supports the entire set of file system commands.

JFS2 is the basic file system type that supports the entire set of file system commands.

A difference between JFS and JFS2 is that JFS2 is designed to support large files and large file systems.

Network File System (NFS)

Is a distributed file system that allows users to access files and directories located on remote computers and use those files and directories as if they were local. For example, users can use operating system commands to create, remove, read, write, and set file attributes for remote files and directories.

CD-ROM File System (CDRFS)

Allows access to the contents of a CD-ROM through the normal file system interfaces (open, read, and close).

DVD-ROM File System (UDFS)

Allows access to the contents of a DVD through the normal file system interfaces.

JFS and JFS2

The journaled file system (JFS) and the enhanced journaled file system (JFS2) are built into the base operating system. Both file system types link their file and directory data to the structure used by the AIX Logical Volume Manager for storage and retrieval.

A difference is that JFS2 is designed to accommodate a 64-bit kernel and larger files.

The following sections describe these file systems. Unless otherwise noted, the following sections apply equally to JFS and JFS2.

JFS and JFS2 functions

Enhanced Journaled File System (JFS2) is a file system that provides the capability to store much larger files than the existing Journaled File System (JFS).

You can choose to implement either JFS or JFS2. JFS2 is the default file system in AIX 6.1.

Note: Unlike the JFS file system, the JFS2 file system will not allow the **link()** API to be used on files of type directory. This limitation may cause some applications that operate correctly on a JFS file system to fail on a JFS2 file system.

The following table provides a summary of JFS and JFS2 functions:

Table 6. JFS and JFS2 functions		
Functions	JFS2	JFS
Fragments and block size	Block sizes (bytes): 512, 1024, 2048, 4096 > Maximum size of the file system with block size of 4096 bytes, when the value of the lff attribute is set to yes , in petabytes (PB): 4 < > Maximum size of the file system when the value of the lff attribute is set to no , in terabytes (TB): 4, 8, 16, 32 < Note: The lff attribute is only supported on a file system with an aggregate block size of 4096 bytes.	Fragment sizes (bytes): 512, 1024, 2048, 4096 Maximum size of the file system in gigabytes (GB): 128, 256, 512, 1024
Maximum file system size	> 4 PB <	1 TB
Minimum file system size	16 MB	Not Applicable
Maximum file size	> 4 PB <	Approximately 63.876 GBs
Number of i-nodes	Dynamic, limited by disk space	Fixed, set at file system creation
Directory organization	B-tree	Linear
Compression	No	Yes
Quotas	Yes	Yes
Error logging	Yes	Yes

Note:

- JFS2 supports the standard AIX error logging scheme. For more information on AIX error logging, please see [Error-Logging Overview](#) in *General Programming Concepts: Writing and Debugging Programs*.
- The maximum size values of the JFS2 files and filesystems that are mentioned in the [Table 6 on page 106](#) are theoretical limits. For currently certified values of the maximum size of JFS2 file and filesystems when you use the **lff=yes** option, refer to the AIX [Release Notes](#).

JFS and JFS2 disk space segmentation

Many UNIX file systems only allocate contiguous disk space in units equal in size to the logical blocks used for the logical division of files and directories. These allocation units are typically referred to as *disk blocks* and a single disk block is used exclusively to store the data contained within a single logical block of a file or directory.

Using a relatively large logical block size (4096 bytes for example) and maintaining disk block allocations that are equal in size to the logical block are advantageous for reducing the number of disk I/O operations that must be performed by a single file system operation. A file or directory data is stored on disk in a small number of large disk blocks rather than in a large number of small disk blocks. For example, a file with a size of 4096 bytes or less is allocated a single 4096-byte disk block if the logical block size is 4096 bytes. A read or write operation therefore only has to perform a single disk I/O operation to access the data on the disk. If the logical block size is smaller requiring more than one allocation for the same amount of data, then more than one disk I/O operation is required to access the data. A large logical block and equal disk block size are also advantageous for reducing the amount of disk space allocation activity that must be performed as new data is added to files and directories, because large disk blocks hold more data.

Restricting the disk space allocation unit to the logical block size can, however, lead to wasted disk space in a file system containing numerous files and directories of a small size. Wasted disk space occurs when a logical block worth of disk space is allocated to a partial logical block of a file or directory. Because partial logical blocks always contain less than a logical block worth of data, a partial logical block only consumes a portion of the disk space allocated to it. The remaining portion remains unused because no other file or directory can write its contents to disk space that has already been allocated. The total amount of wasted disk space can be large for file systems containing a large number of small files and directories.

The journaled file system (JFS) divides disk space into allocation units called *fragments*. The enhanced journaled file system (JFS2) segments disk space into *blocks*. The objective is the same: to efficiently store data.

JFS fragments are smaller than the default disk allocation size of 4096 bytes. Fragments minimize wasted disk space by more efficiently storing the data in a file or directory partial logical blocks. The functional behavior of JFS fragment support is based on that provided by Berkeley Software Distribution (BSD) fragment support.

JFS2 supports multiple file system block sizes of 512, 1024, 2048, and 4096. Smaller block sizes minimize wasted disk space by more efficiently storing the data in a file or directory's partial logical blocks. Smaller block sizes also result in additional operational overhead. The block size for a JFS2 is specified during its creation. Different file systems can have different block sizes, but only one block size can be used within a single file system.

JFS fragments

In JFS, the disk space allocation unit is called a *fragment*, and it can be smaller than the logical block size of 4096 bytes.

With the use of fragments smaller than 4096 bytes, the data contained within a partial logical block can be stored more efficiently by using only as many fragments as are required to hold the data. For example, a partial logical block that only has 500 bytes could be allocated a fragment of 512 bytes (assuming a fragment size of 512 bytes), thus greatly reducing the amount of wasted disk space. If the storage requirements of a partial logical block increase, one or more additional fragments are allocated.

The fragment size for a file system is specified during its creation. The allowable fragment sizes for journaled file systems (JFS) are 512, 1024, 2048, and 4096 bytes. Different file systems can have different fragment sizes, but only one fragment size can be used within a single file system. Different

fragment sizes can also coexist on a single system (machine) so that users can select a fragment size most appropriate for each file system.

JFS fragment support provides a view of the file system as a contiguous series of fragments rather than as a contiguous series of disk blocks. To maintain the efficiency of disk operations, however, disk space is often allocated in units of 4096 bytes so that the disk blocks or allocation units remain equal in size to the logical blocks. A disk-block allocation in this case can be viewed as an allocation of 4096 bytes of contiguous fragments.

Both operational overhead (additional disk seeks, data transfers, and allocation activity) and better utilization of disk space increase as the fragment size for a file system decreases. To maintain the optimum balance between increased overhead and increased usable disk space, the following factors apply to JFS fragment support:

- Where possible, disk space allocations of 4096 bytes of fragments are maintained for a file or the logical blocks of a directory.
- Only partial logical blocks for files or directories less than 32KB in size can be allocated less than 4096 bytes of fragments.

As the files and directories within a file system grow beyond 32 KB in size, the benefit of maintaining disk space allocations of less than 4096 bytes for partial logical blocks diminishes. The disk space savings as a percentage of total file system space grows small while the extra performance cost of maintaining small disk space allocations remains constant. Since disk space allocations of less than 4096 bytes provide the most effective disk space utilization when used with small files and directories, the logical blocks of files and directories equal to or greater than 32 KB are always allocated 4096 bytes of fragments. Any partial logical block associated with such a large file or directory is also allocated 4096 bytes of fragments.

JFS2 blocks

The enhanced journaled file system segments disk space into *blocks*. JFS2 supports multiple file system block sizes of 512, 1024, 2048, and 4096.

Different file systems can have different block sizes, but only one block size can be used within a single file system.

Smaller block sizes minimize wasted disk space by more efficiently storing the data in a file or directory's partial logical blocks. Smaller block sizes can also result in additional operational overhead. Also, device drivers must provide disk block addressability that is the same or smaller than the file system block size.

Because disk space is allocated in smaller units for a file system with a block size other than 4096 bytes, allocation activity can occur more often when files or directories are repeatedly extended in size. For example, a write operation that extends the size of a zero-length file by 512 bytes results in the allocation of one block to the file, assuming a block size of 512 bytes. If the file size is extended further by another write of 512 bytes, an additional block must be allocated to the file. Applying this example to a file system with 4096-byte blocks, disk space allocation occurs only once, as part of the first write operation. No additional allocation activity is performed as part of the second write operation since the initial 4096-byte block allocation is large enough to hold the data added by the second write operation.

File system block size is specified during the file system's creation with the System Management Interface Tool (SMIT), or the **crfs** and **mkfs** commands. The decision of which file system block size to choose should be based on the projected size of files contained by the file system.

The file system block size value can be identified through the System Management Interface Tool (SMIT), or the **lsfs** command. For application programs, the **statfs** subroutine can be used to identify the file system block size.

Blocks serve as the basic unit of disk space allocation, and the allocation state of each block within a file system is recorded in the file system block allocation map. More virtual memory and file system disk space might be required to hold block allocation maps for file systems with a block size smaller than 4096 bytes.

Variable number of i-nodes

Segmenting disk space at sizes smaller than 4096 bytes optimizes disk space utilization, but it increases the number of small files and directories that can be stored within a file system.

However, disk space is only one of the file system resources required by files and directories: each file or directory also requires a disk i-node.

JFS and i-nodes

JFS allows you to specify the number of disk i-nodes created within a file system in case more or fewer than the default number of disk i-nodes is desired.

The number of disk i-nodes at file system creation is specified in a value called as the *number of bytes per i-node* or *NBPI*. For example, an NBPI value of 1024 causes a disk i-node to be created for every 1024 bytes of file system disk space. Another way to look at this is that a small NBPI value (512 for instance) results in a large number of i-nodes, while a large NBPI value (such as 16,384) results in a small number of i-nodes.

For JFS file systems, one i-node is created for every NBPI bytes of allocation group space allocated to the file system. The total number of i-nodes in a file system limits the total number of files and the total size of the file system. An allocation group can be partially allocated, though the full number of i-nodes per allocation group is still allocated. NBPI is inversely proportional to the total number of i-nodes in a file system.

The JFS restricts all file systems to 16M (2^{24}) i-nodes.

The set of allowable NBPI values vary according to the allocation group size (*agsize*). The default is 8 MB. The allowable NBPI values are 512, 1024, 2048, 4096, 8192, and 16,384 with an *agsize* of 8 MB. A larger *agsize* can be used. The allowable values for *agsize* are 8, 16, 32, and 64. The range of allowable NBPI values scales up as *agsize* increases. If the *agsize* is doubled to 16 MB, the range of NBPI values also double: 1024, 2048, 4096, 8193, 16384, and 32768.

Fragment size and the NBPI value are specified during the file system's creation with the System Management Interface Tool (SMIT), or the **crfs** and **mkfs** commands. The decision of fragment size and how many i-nodes to create for the file system is based on the projected number and size of files contained by the file system.

You can identify fragment size and NBPI value using the System Management Interface Tool (SMIT), or the **lsfs** command. For application programs, use the `statfs` subroutine to identify the file system fragment size.

JFS2 and i-nodes

JFS2 allocates i-nodes as needed.

If there is room in the file system for additional i-nodes, they are automatically allocated. Therefore, the number of i-nodes available is limited by the size of the file system itself.

JFS and JFS2 size limitations

> You can define the maximum size for a journaled file system (JFS) or the enhanced journaled file system (JFS2), when you create the file system. The decision of what size to define for a JFS or JFS2 is based on several significant issues.<

> For a JFS2 file system, a block size of 4096 bytes is allowed. If the value of the **lff** attribute is set to **no**, the maximum size of a file system with block size of 4096 bytes is 32 TB. If the value of the **lff** attribute is set to **yes**, the maximum size of a file system with block size of 4096 bytes is 4 PB. The minimum size of a file system for a JFS2 file system is 16 MB.<

Although file systems that use allocation units smaller than 4096 bytes require substantially less disk space than those using the default allocation unit of 4096 bytes, the use of smaller fragments might incur performance costs.

The allocation state of each fragment (JFS) or block (JFS2) within a file system is recorded in the file system allocation map. More virtual memory and file system disk space might be required to hold allocation maps for file systems with a fragment or block size smaller than 4096 bytes.

Because disk space is allocated in smaller units for a file system with a fragment (JFS) or block (JFS2) size other than 4096 bytes, allocation activity can occur more often when files or directories are repeatedly extended in size. For example, a write operation that extends the size of a zero-length file by 512 bytes results in the allocation of one 512-byte fragment or block to the file, depending on the file system type. If the file size is extended further by another write of 512 bytes, an additional fragment or block must be allocated to the file. Applying this example to a file system with 4096-byte fragments or blocks, disk space allocation occurs only once, as part of the first write operation. No additional allocation activity must be performed as part of the second write operation since the initial 4096-byte allocation is large enough to hold the data added by the second write operation. Allocation activity can be minimized if the files are extended by 4096 bytes at a time.

One size-related issue is the size of the file system log.

For JFS, in most instances, multiple file systems use a common log configured to be 4 MB in size. For example, after initial installation, all file systems within the root volume group use logical volume hd8 as a common JFS log. The default logical volume partition size is 4 MB, and the default log size is one partition, therefore, the root volume group normally contains a 4 MB JFS log. When file systems exceed 2 GB or when the total amount of file system space using a single log exceeds 2 GB, the default log size might not be sufficient. In either case, the log sizes are scaled upward as the file system size increases. When the size of the log logical volume is changed, the **logform** command must be run to reinitialize the log before the new space can be used. The JFS log is limited to a maximum size of 256 MB.

There is a practical limit to the size of the combined file systems that a single JFS log can support. As a guideline, one trillion bytes of total file system capacity is the recommended limitation for a single JFS log. When this guideline is exceeded or is close to being exceeded, or when out-of-memory errors occur from the **logredo** command (which is called by the **fsck** command), add an additional JFS log and then share the load between the two JFS log files.

For JFS2, in most instances, multiple file systems also use a common log. When file systems exceed 2 GB or when the total amount of file system space using a single log exceeds 2 GB, the default log size might not be sufficient. In either case, you can scale log sizes upward as the file system size increases or you can add an additional JFS2 log and then share the load between the two JFS2 log files.

JFS size limits

The maximum JFS size is defined when the file system is created. The NBPI, fragment size, and allocation group size are contributing factors to the decision.

The file system size limitation is the minimum of the following:

$$NBPI * 2^{24}$$

or

$$FragmentSize * 2^{28}$$

For example, if you select an NBPI ratio of 512, the file system size is limit to 8 GB ($512 * 2^{24} = 8 \text{ GB}$). JFS supports NBPI values of 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, and 131072.

The JFS restricts all file systems to 16M (2^{24}) i-nodes.

One i-node is created for every *NBPI* bytes of allocation group space allocated to the file system. An allocation group can be partially allocated, though the full number of i-nodes per allocation group is still allocated. NBPI is inversely proportional to the total number of i-nodes in a file system.

The JFS segregates file system space into groupings of i-nodes and disk blocks for user data. These groupings are called allocation groups. The allocation group size can be specified when the file system is created. The allocation group sizes are 8M, 16M, 32M, and 64M. Each allocation group size has an associated NBPI range. The ranges are defined by the following table:

Allocation Group Size in Megabytes	Allowable NBPI Values
8	512, 1024, 2048, 4096, 8192, 16384
16	1024, 2048, 4096, 8192, 16384, 32768
32	2048, 4096, 8192, 16384, 32768, 65536
64	4096, 8192, 16384, 32768, 65536, 131072

The JFS supports four fragment sizes of 512, 1024, 2048, and 4096 byte units of contiguous disk space. The JFS maintains fragment addresses in i-nodes and indirect blocks as 28-bit numbers. Each fragment must be addressable by a number from 0 to (2^{28}) .

JFS2 size limits

Extremely large JFS2 file systems that contain very large files are more practical to maintain than those that contain a large number of small files. When a large file system contains many small files, the **fsck** command and other file system maintenance tasks take a long time to run.

The size limitations of the JFS2 file and file system depend on the value of the **lff** filesystem attribute:

Item	> lff=yes <	> lff=no <
Maximum size of a JFS2 file system:	> 4 PB <	> 32 TB <
Maximum size of a JFS2 file:	> 4 PB <	> 16 TB <
Minimum size of a JFS2 file system:	> 16 MB <	> 16 MB <

JFS free space fragmentation

For JFS file systems, using fragments smaller than 4096 bytes can cause greater fragmentation of the free space on the disk.

For example, consider an area of the disk that is divided into eight fragments of 512 bytes each. Suppose that different files, requiring 512 bytes each, have written to the first, fourth, fifth, and seventh fragments in this area of the disk, leaving the second, third, sixth, and eighth fragments free. Although four fragments representing 2048 bytes of disk space are free, no partial logical block requiring four fragments (or 2048 bytes) is allocated for these free fragments, since the fragments in a single allocation must be contiguous.

Because the fragments allocated for a file or directory logical blocks must be contiguous, free space fragmentation can cause a file system operation that requests new disk space to fail even though the total amount of available free space is large enough to satisfy the operation. For example, a write operation that extends a zero-length file by one logical block requires 4096 bytes of contiguous disk space to be allocated. If the file system free space is fragmented and consists of 32 noncontiguous 512-byte fragments or a total of 16 KB of free disk space, the write operation will fail because eight contiguous fragments (or 4096 bytes of contiguous disk space) are not available to satisfy the write operation.

A JFS file system with an unmanageable amount of fragmented free space can be defragmented with the **defragfs** command. Running the **defragfs** command has a positive impact on performance.

Sparse files

A file is a sequence of indexed blocks. Blocks are mapped from the i-node to the logical offset of the file they represent.

A file that has one or more indexes that are not mapped to a data block is referred to as being *sparsely-allocated* or a *sparse file*. A sparse file will have a size associated with it, but it will not have all of the

data blocks allocated to fulfill the size requirements. To identify if a file is sparsely-allocated, use the **fileplace** command. It will indicate all blocks in the file that are not currently allocated.

Note: In most circumstances, **du** can also be used to determine if the number of data blocks allocated to a file do not match those required to hold a file of its size. A compressed file system might show the same behavior for files that are not sparsely-allocated.

A sparse file is created when an application extends a file by seeking to a location outside the currently allocated indexes, but the data that is written does not occupy all of the newly assigned indexes. The new file size reflects the farthest write into the file.

A read to a section of a file that has unallocated data blocks results in a buffer of zeroes being returned. A write to a section of a file that has unallocated data blocks causes the necessary data blocks to be allocated and the data written.

This behavior can affect file manipulation or archival commands. For example, the following commands do not preserve the sparse allocation of a file:

- **cp**
- **mv**
- **tar**
- **cpio**

Note: In the case of **mv**, this only applies to moving a file to another file system. If the file is moved within the same file system, it will remain sparse.

The result of a file being copied or restored from the preceding commands has each data block allocated, and thus have no sparse characteristics. However, the following archival commands either preserve sparse characteristics or actively sparse a file:

- **backup**
- **restore**
- **pax**

Because it is possible to overcommit the resources of a file system with sparse files, care should be taken in the use and maintenance of files of this type.

JFS and large files

You can create large files with the JFS file system type.

All JFS2 file systems support large files.

File systems enabled for large files can be created with the **crfs** and **mkfs** commands. Both commands have an option (**bf=true**) to specify file systems enabled for large files. You can also use SMIT to create these file systems.

In file systems enabled for large files, file data stored before the 4 MB file offset is allocated in 4096-byte blocks. File data stored beyond the 4 MB file offset is allocated with large disk blocks of 128 KB in size. The large disk blocks are actually 32 contiguous 4096-byte blocks.

For example, in a regular file system, the 132-MB file requires 33K 4-KB disk blocks (33 single indirect blocks each filled with 1024 4 KB disk addresses). A 132-MB file in a file system enabled for large files has 1024 4-KB disk blocks and 1024 128-KB disk blocks. The large file geometry requires only two single indirect blocks for the 132 MB file. Both large and regular file types require one double indirect block.

Large disk blocks require 32 contiguous 4 KB blocks. If you write to large files beyond the 4 MB, file offset will fail with ENOSPC if the file system does not contain 32 unused contiguous 4 KB blocks.

Note: The file system may have thousands of free blocks, but if 32 of them are not contiguous, the allocation will fail.

The **defragfs** command reorganizes disk blocks to provide larger contiguous free block areas.

The JFS is required to initialize all new disk allocations. The JFS starts the kernel kproc procedure used to zero initial file allocations when mounting the first large file enabled file system on your system. The kproc procedure remains once the file system enabled for large files has successfully unmounted.

JFS data compression

JFS supports fragmented and compressed file systems, which save disk space by allowing a logical block to be stored on the disk in units or "fragments" smaller than the full block size of 4096 bytes.

Data compression is not supported for JFS2.

In a fragmented file system, only the last logical block of files no larger than 32KB are stored in this manner, so that fragment support is only beneficial for file systems containing numerous small files. Data compression, however, allows all logical blocks of any-sized file to be stored as one or more contiguous fragments. On average, data compression saves disk space by about a factor of two.

The use of fragments and data compression does, however, increase the potential for fragmentation of the disk free space. Fragments allocated to a logical block must be contiguous on the disk. A file system experiencing free space fragmentation might have difficulty locating enough contiguous fragments for a logical block allocation, even though the total number of free fragments may exceed the logical block requirements. The JFS alleviates free space fragmentation by providing the **defragfs** program which "defragments" a file system by increasing the amount of contiguous free space. This utility can be used for fragmented and compressed file systems. The disk space savings gained from fragments and data compression can be substantial, while the problem of free space fragmentation remains manageable.

Data compression in the current JFS is compatible with previous versions of this operating system. The API comprised of all the system calls remains the same in both versions of the JFS.

JFS data compression implementation

Data compression is an attribute of a file system which is specified when the file system is created with the **crfs** or **mkfs** command. You can use SMIT to specify data compression.



Attention: The root file system (/) must not be compressed. Compressing the /usr file system is not recommended because **installp** must be able to accurately calculate its size for updates and new installs.

Compression only applies to regular files and long symbolic links in such file systems. Fragment support continues to apply to directories and metadata that are not compressed. Each logical block of a file is compressed by itself before being written to the disk. Compression in this manner facilitates random seeks and updates, while losing only a small amount of freed disk space in comparison to compressing data in larger units.

After compression, a logical block usually requires less than 4096 bytes of disk space. The compressed logical block is written to the disk and allocated only the number of contiguous fragments required for its storage. If a logical block does not compress, then it is written to disk in its uncompressed form and allocated 4096 bytes of contiguous fragments.

The **lsfs -q** command displays the current value for compression. You can also use the SMIT to identify data compression.

Implicit behavior of JFS data compression

Because a program that writes a file does not expect an out-of-space (ENOSPC) condition to occur after a successful write (or successful store for mapped files), it is necessary to guarantee that space be available when logical blocks are written to the disk.

This is accomplished by allocating 4096 bytes to a logical block when it is first modified so that there is disk space available even if the block does not compress. If a 4096-byte allocation is not available, the system returns an ENOSPC or EDQUOT error condition even though there might be enough disk space to accommodate the compressed logical block. Premature reporting of an out-of-space condition is most likely when operating near disk quota limits or with a nearly full file system.

Compressed file systems might also exhibit the following behavior:

- Because 4096 bytes are initially allocated to a logical block, certain system calls might receive an ENOSPC or EDQUOT error. For example, an old file might be mapped using the `mmap` system call, and a store operation into a previously written location can result in an ENOSPC error.
- With data compression, a full disk block remains allocated to a modified block until it is written to disk. If the block had a previously committed allocation of less than a full block, the amount of disk space tied up by the block is the sum of the two, the previous allocation not being freed until the file (i-node) is committed. This is the case for normal fragments. The number of logical blocks in a file that can have previously committed allocations is, at most, one for normal fragments but can be as many as the number of blocks in a file with compression.
- None of the previously committed resources for a logical block are freed until the `fsync` or `sync` system call is run by the application program.
- The `stat` system call indicates the number of fragments allocated to a file. The number reported is based on 4096 bytes being allocated to modified but unwritten blocks and the compressed size of unmodified blocks. Previously committed resources are not counted by the `stat` system call. The `stat` system call reports the correct number of allocated fragments after an i-node commit operation if none of the modified blocks compressed. Similarly, disk quotas are charged for the current allocation. As the logical blocks of a file are written to the disk, the number of fragments allocated to them decrease if they compress, and thereby change disk quotas and the result from `stat`.

JFS data compression algorithm

The compression algorithm is an IBM version of LZ. In general, LZ algorithms compress data by representing the second and later occurrences of a given string with a pointer that identifies the location of the first occurrence of the string and its length.

At the beginning of the compression process, no strings have been identified, so at least the first byte of data must be represented as a "raw" character requiring 9-bits (0,byte). After a given amount of data is compressed, say N bytes, the compressor searches for the longest string in the N bytes that matches the string starting at the next unprocessed byte. If the longest match has length 0 or 1, the next byte is encoded as a "raw" character. Otherwise, the string is represented as a (pointer,length) pair and the number of bytes processed is incremented by length. Architecturally, IBM LZ supports values of N of 512, 1024, or 2048. IBM LZ specifies the encoding of (pointer,length) pairs and of raw characters. The pointer is a fixed-length field of size $\log_2 N$, while the length is encoded as a variable-length field.

Performance costs of JFS data compression

Because data compression is an extension of fragment support, the performance associated with fragments also applies to data compression.

Compressed file systems also affect performance in the following ways:

- It can require a great deal of time to compress and extract data so that the usability of a compressed file system might be limited for some user environments.
- Most UNIX regular files are written only once, but some are updated in place. For the latter, data compression has the additional performance cost of having to allocate 4096 bytes of disk space when a logical block is first modified, and then reallocate disk space after the logical block is written to the disk. This additional allocation activity is not necessary for regular files in a uncompressed file system.
- Data compression increases the number of processor cycles. For the software compressor, the number of cycles for compression is approximately 50 cycles per byte, and for decompression 10 cycles per byte.

JFS online backups and JFS2 snapshots

You can make a point-in-time image of a JFS file system or of a JFS2 file system, which you can then use for backup purposes. There are differences, however, in the requirements and behavior of this image for each file system type.

For a JFS file system, you can split off a read-only static copy of a mirrored copy of the file system. Typically, a mirrored copy is updated whenever the original file system is updated, but this point-in-time copy does not change. It remains a stable image of the point in time at which the copy was made. When this image is used for backing up, any modifications that begin after you begin the procedure to create the

image might not be present in the backup copy. Therefore, it is recommended that file system activity be minimal while the split is taking place. Any changes that occur after the split is made will not be present in the backup copy.

For a JFS2 file system, the point-in-time image is called a *snapshot*. The snapshot remains static and it retains the same security permissions as the original file system (called the *snappedFS*) had when the snapshot was made. Also, you can create a JFS2 snapshot without unmounting or quiescing the file system. You can use a JFS2 snapshot to use as an online backup of the file system, to access the files or directories as they existed when the snapshot was taken, or to back up to removable media. Note the following about JFS2 snapshots:

- A snapshot image of the root (/) or /usr file system is overwritten when the system is rebooted. Snapshots of other file systems can be preserved by unmounting the file system before rebooting. Snapshots created in AIX 5.2 with 5200-01 are recoverable. When **fsck** or **logredo** runs on a JFS2 filesystem with a snapshot created on AIX 5.2 with 5200-01, the snapshot will be preserved. A cleanly unmounted filesystem with an AIX 5.2-created snapshot will also be recoverable once it is mounted on an AIX 5.2 with 5200-01 system.
- Running the **defragfs** command against a file system with snapshots is not recommended. Every block that is moved during defragmentation must also be copied to the snapshot, which is both time consuming and a waste of space in the snapshot logical volume.
- If a snapshot runs out of space, all snapshots for that snappedFS are deleted. This failure writes an entry to the error log.
- If a write to a snapshot fails, all snapshots for that snappedFS are deleted. This failure writes an entry to the error log.
- A snapshot that is created or accessed on a AIX 5.2 with 5200-01 system cannot be accessed on an AIX 5.2 system. These snapshots must be deleted before the filesystem can be mounted.
- A JFS2 file system that has a snapshot on AIX 5.3 cannot be accessed on any releases prior to AIX 5.2 with 5200-01. If the system is to be moved back, the snapshots must be deleted first to allow file system access.

JFS online backups

You can make a point-in-time image of a JFS file system that you can then use for backup purposes.

For a JFS file system, you can split off a read-only static copy of a mirrored copy of the file system. Typically, a mirrored copy is updated whenever the original file system is updated, but this point-in-time copy does not change. It remains a stable image of the point in time at which the copy was made. When this image is used for backing up, any modifications that begin after you begin the procedure to create the image might not be present in the backup copy. Therefore, it is recommended that file system activity be minimal while the split is taking place. Any changes that occur after the split is made will not be present in the backup copy.

JFS2 snapshots

You can make a point-in-time image of a JFS2 file system that you can then use for backup purposes.

The point-in-time image for a JFS2 file system is called a *snapshot*. The snapshot remains static and retains the same security permissions that the original file system (called the *snappedFS*) had when the snapshot was made. Also, you can create a JFS2 snapshot without unmounting the file system, or quiescing the file system. You can use a JFS2 snapshot to:

- Access the files or directories as they existed when the snapshot was taken.
- Backup to removable media.

There are two types of JFS2 snapshots: internal and external. A JFS2 external snapshot is created in a separate logical volume from the file system. The external snapshot can be mounted separately from the file system at its own unique mount point.

A JFS2 internal snapshot is created in the same logical volume as the file system and allocates blocks from the file system. An internal snapshot is accessible from the invisible `.snapshot` directory in the

root of the JFS2 file system with the snapshot. A JFS2 file system must be enabled to support internal snapshots at the time the file system is created.

JFS2 snapshots do not support checking of file system quotas. You cannot use the **repquota** command on the snapshot to determine the state of the quota. The point-in-time quota information is preserved if you roll back the file system image to the snapshot image. Note the following considerations specific to JFS2 external snapshots and JFS2 internal snapshots:

- An external snapshot that is created or accessed on an AIX 5.2 with 5200-01 system cannot be accessed on an AIX 5.2 system. These snapshots must be deleted before the file system can be mounted.
- A JFS2 file system that has a snapshot on AIX 5.3 cannot be accessed on any releases prior to AIX 5.2 with 5200-01. If the system is to be moved back, the snapshots must be deleted first to allow file system access.
- Running the **defragfs** command against a JFS2 file system with external snapshots is not recommended because every block that is moved during defragmentation must also be copied to the snapshot, which is both time consuming and a waste of space in the snapshot logical volume.
- If an external snapshot runs out of space, or if an external snapshot fails, all external snapshots for that snappedFS are marked invalid. Further access to the snapshot will fail. These failures write an entry to the error log.

Internal JFS2 snapshot considerations:

- Internal snapshots are preserved when the **logredo** command runs on a JFS2 file system with an internal snapshot.
- Internal snapshots are removed if the **fsck** command has to modify a JFS2 file system to repair it.
- If an internal snapshot runs out of space, or if a write to an internal snapshot fails, all internal snapshots for that snappedFS are marked invalid. Further access to the internal snapshots will fail. These failures write an entry to the error log.
- Internal snapshots are not separately mountable. You can access internal snapshots in the `.snapshot` directory of the root of the file system immediately after they are created. Consequently, you can access internal snapshots through an NFS server without having to export a separate mount point for the snapshot.
- Internal snapshots are not compatible with AIX releases prior to AIX 6.1. A JFS2 file system created to support internal snapshots cannot be modified on an earlier release of AIX.
- A JFS2 file system created to support internal snapshots is also enabled to support Extended Attributes Version 2.
- A JFS2 file system with internal snapshots cannot be used with a Data Management Application Programming Interface (DMAPI).
- You cannot use the **defragfs** command against a JFS2 file system with internal snapshots.
- The `.snapshot` directory is not returned from the **readdir()** system call. This prevents unintended visiting of the snapshots. Any system calls or commands that depend on the **readdir()** system call fail with the `.snapshot` directory (for example, the **/bin/pwd** command and the **getcwd()** system call of the `.snapshot` directory cannot find the parent directory).

Compatibility and migration

JFS file systems are fully compatible within AIX 5.1 and AIX 5.2. Previous supported versions of the operating system are compatible with the current JFS, although file systems with a nondefault fragment size, NBPI value, or allocation group size might require special attention if migrated to a previous version.

Note: JFS file systems are not supported on disks that have a sector size of 4 KB. Therefore, when you create a file system or perform backup operations, ensure that the disks are not of 4 KB sector size.

JFS2 file systems, with the exception of snapshots, are compatible within AIX 5.1 and AIX 5.2, but not with earlier versions of the operating system. JFS2 file systems with snapshots are not supported in AIX 5.1. Always ensure a clean unmount of all JFS2 file systems before reverting to an earlier version of AIX because the **logredo** command does not necessarily run against a file system created for a later release.

Note: JFS2 file systems created or converted to v2 format cannot be accessed on prior releases of AIX.

The following list describes aspects that might cause problems with file systems created under earlier versions of the operating system:

JFS file system images

Any JFS file system image created with the default fragment size and NBPI value of 4096 bytes, and default allocation group size (agsize) of 8 can be interchanged with JFS file system images created under AIX 4.3 and later versions of this operating system without requiring any special migration activities.

Note: JFS2 Snapshots: JFS2 snapshots created or accessed in AIX 5L Version 5.2 with the 5200-01 Recommended Maintenance package are not accessible on earlier releases. These snapshots must be deleted before the filesystem can be mounted.

Backup and restore between JFS file systems

Backup and restore sequences can be performed between JFS file systems with different block sizes, however because of increased disk utilization, restore operations can fail due to a lack of free blocks if the block size of the source file system is smaller than the block size of the target file system. This is of particular interest for full file system backup and restore sequences and can even occur when the total file system size of the target file system is larger than that of the source file system.

While backup and restore sequences can be performed from compressed to uncompressed file systems or between compressed file systems with different fragment sizes, because of the enhanced disk utilization of compressed file systems, restore operations might fail due to a shortage of disk space. This is of particular interest for full file system backup and restore sequences and might even occur when the total file system size of the target file system is larger than that of the source file system.

JFS and JFS2 device driver limitations

A device driver must provide disk block addressability that is the same or smaller than the JFS file system fragment size or the JFS2 block size. For example, if a JFS file system was made on a user supplied RAM disk device driver, the driver must allow 512 byte blocks to contain a file system that had 512 byte fragments. If the driver only allowed page level addressability, a JFS with a fragment size of 4096 bytes could only be used.

Copying a JFS to another physical volume

You can copy a JFS file system to a different physical volume while retaining file system integrity.

The following scenario describes copying JFS file system to a different physical volume while retaining file system integrity.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

To copy a JFS to another physical volume while maintaining file system integrity, do the following:

1. Stop activity for the file system that you want to copy. Unless the application that is using the file system is quiescent or the file system is in a state that is known to you, you cannot know what is in the data of the backup.
2. Mirror the logical volume, by typing the following SMIT fast path on the command line:

```
smit mklvcopy
```

3. Copy the file system, using the following command:

```
chfs -a splitcopy=/backup -a copy=2 /testfs
```

The **splitcopy** parameter for the **-a** flag causes the command to split off a mirrored copy of the file system and mount it read-only at the new mount point. This action provides a copy of the file system with consistent journaled meta data that can be used for backup purposes.

4. If you want to move the mirrored copy to a different mount point, use the following SMIT fast path:

At this point, the file system copy is usable.

CD-ROM file system and UDF file system

The CD-ROM file system (CDRFS) is a read-only local file system implementation that might be stored on CD-ROM media, CD-RW media if write protected, and DVD-ROM media. The maximum CDRFS file size is 2 GB, regardless of the media used. The Universal Disk Format (UDF) file system is a writeable local file system implementation that might be stored as read-only on DVD-ROM media, or as read-write in DVD-RAM media.

CDs are automatically mounted by default but this feature can be disabled. If the feature has been disabled, use the **cdmount** command to mount the CDRFS file system.

AIX supports the following CDRFS volume and file structure formats:

Type	Description
The ISO 9660:1988(E) standard	The CDRFS supports ISO 9660 level 3 of interchange and level 1 of implementation.
The High Sierra Group Specification	Precedes the ISO 9660 and provides backward compatibility with previous CD-ROMs.
The Rock Ridge Group Protocol	Specifies extensions to the ISO 9660 that are fully compliant with the ISO 9660 standard, and that provide full POSIX file system semantics based on the System Use Sharing Protocol (SUSP) and the Rock Ridge Interchange Protocol (RRIP), enabling mount/access CD-ROM as with any other UNIX file system.
The CD-ROM eXtended Architecture File Format (in Mode 2 Form 1 sector format only)	The CD-ROM eXtended Architecture (XA) file format specifies extensions to the ISO 9660 that are used in CD-ROM-based multimedia applications for example, Photo CD.

For all volume and file structure formats, the following restrictions apply:

- Single-volume volume set only
- Non-interleaved files only

The CDRFS is dependent upon the underlying CD-ROM device driver to provide transparency of the physical sector format (CD-ROM Mode 1 and CD-ROM XA Mode 2 Form 1), and the multisession format of the disks (mapping the volume descriptor set from the volume recognition area of the last session).

Note: The CDRFS must be unmounted from the system before you can remove the CD-ROM media.

There is another supported file system type called UDFS, which is a read-only file system stored on DVD-ROM media. UDFS must be unmounted from the system before you can remove the media. AIX supports UDFS format versions 1.50, 2.00, and 2.01.

UDFS must be exported using NFS in read-only mode. Writing to an NFS mounted UDFS is not supported.

To use the **cdmount** command to automatically mount a read/write UDFS, edit the `cdromd.conf` file. You can also manually mount a read/write UDFS with the **mount** command.

Directories

A *directory* is a unique type of file that contains only the information needed to access files or other directories. As a result, a directory occupies less space than other types of files.

File systems consist of groups of directories and the files within the directories. File systems are commonly represented as an inverted tree. The root directory, denoted by the slash (/) symbol, defines a file system and appears at the top of a file system tree diagram.

Directories branch downward from the root directory in the tree diagram and can contain both files and subdirectories. Branching creates unique paths through the directory structure to every object in the file system.

Collections of files are stored in directories. These collections of files are often related to each other; storing them in a structure of directories keeps them organized.

A *file* is a collection of data that can be read from or written to. A file can be a program you create, text you write, data you acquire, or a device you use. Commands, printers, terminals, correspondence, and application programs are all stored in files. This allows users to access diverse elements of the system in a uniform way and gives great flexibility to the file system.

Directories let you group files and other directories to organize the file system into a modular hierarchy, which gives the file system structure flexibility and depth.

Directories contain directory entries. Each entry contains a file or subdirectory name and an index node reference number (*i-node* number). To increase speed and enhance use of disk space, the data in a file is stored at various locations in the memory of the computer. The i-node number contains the addresses used to locate all the scattered blocks of data associated with a file. The i-node number also records other information about the file, including times of modification and access, access modes, number of links, file owner, and file type.

A special set of commands controls directories. For example, you can link several names for a file to the same i-node number by creating directory entries with the **ln** command.

Because directories often contain information that should not be available to all users of the system, directory access can be protected. By setting a directory's permissions, you can control who has access to the directory, also determining which users (if any) can alter information within the directory.

Types of directories

Directories can be defined by the operating system, by the system administrator, or by users.

The system-defined directories contain specific kinds of system files, such as commands. At the top of the file system hierarchy is the system-defined / (root) directory. The / (root) directory usually contains the following standard system-related directories:

Item	Description
/dev	Contains special files for I/O devices.
/etc	Contains files for system initialization and system management.
/home	Contains login directories for the system users.
/tmp	Contains files that are temporary and are automatically deleted after a specified number of days.
/usr	Contains the lpp, include, and other system directories.
/usr/bin	Contains user-executable programs.

Some directories, such as your login or home directory (\$HOME), are defined and customized by the system administrator. When you log in to the operating system, the login directory is the current directory.

Directories that you create are called *user-defined* directories. These directories allow you to organize and maintain your files.

Directory organization

Directories contain files, subdirectories, or a combination of both. A *subdirectory* is a directory within a directory. The directory containing the subdirectory is called the *parent directory*.

Each directory has an entry for the parent directory in which it was created, `..` (dot dot), and an entry for the directory itself, `.` (dot). In most directory listings, these files are hidden.

Directory Tree

The file system structure of directories can easily become complex. Attempt to keep the file and directory structure as simple as possible. Create files and directories with easily recognizable names. This makes working with files easier.

Parent Directory

Each directory, except for `/` (`root`), has one parent directory and may have child directories.

Home Directory

When you log in, the system puts you in a directory called your *home directory* or login directory. Such a directory is set up by the system administrator for each user. Your home directory is the repository for your personal files. Normally, directories that you create for your own use will be subdirectories of your home directory. To return to your home directory at any time, type the `cd` command and press Enter at the prompt.

Working Directory

You are always working within a directory. Whichever directory you are currently working in is called your *current* or *working* directory. The `pwd` (present working directory) command reports the name of your working directory. Use the `cd` command to change working directories.

Directory naming conventions

The name of each directory must be unique within the directory where it is stored. This ensures that the directory has a unique path name in the file system.

Directories follow the same naming conventions as files, as explained in [Filename naming conventions](#).

Directory abbreviations

Abbreviations provide a convenient way to specify certain directories.

The following is a list of abbreviations:

Abbreviation	Meaning
<code>.</code>	The current working directory.
<code>..</code>	The directory above the current working directory (the parent of the current directory).
<code>~</code>	Your home directory. (This is not true for the Bourne shell. For more information, see Bourne Shell .)
<code>\$HOME</code>	Your home directory. (This is true for all shells.)

Directory path names

Each file and directory can be reached by a unique path, known as the *path name*, through the file system tree structure. The path name specifies the location of a directory or file within the file system.

Note: Path names cannot exceed 1023 characters in length.

The file system uses the following kinds of path names:

Item	Description
absolute path name	Traces the path from the <code>/</code> (<code>root</code>) directory. Absolute path names always begin with the slash (<code>/</code>) symbol.

Item	Description
relative path name	Traces the path from the current directory through its parent or its subdirectories and files.

An absolute path name represents the complete name of a directory or file from the `/` (root) directory downward. Regardless of where you are working in the file system, you can always find a directory or file by specifying its absolute path name. Absolute path names start with a slash (`/`), the symbol representing the root directory. The path name `/A/D/9` is the absolute path name for 9. The first slash (`/`) represents the `/` (root) directory, which is the starting place for the search. The remainder of the path name directs the search to A, then to D, and finally to 9.

Two files named 9 can exist because the absolute path names to the files give each file a unique name within the file system. The path names `/A/D/9` and `/C/E/G/9` specify two unique files named 9.

Unlike full path names, relative path names specify a directory or file based on the current working directory. For relative path names, you can use the notation dot dot (`..`) to move upward in the file system hierarchy. The dot dot (`..`) represents the parent directory. Because relative path names specify a path starting in the current directory, they do not begin with a slash (`/`). Relative path names are used to specify the name of a file in the current directory or the path name of a file or directory above or below the level of the current directory in the file system. If D is the current directory, the relative path name for accessing 10 is `F/10`. However, the absolute path name is always `/A/D/F/10`. Also, the relative path name for accessing 3 is `../.. /B/3`.

You can also represent the name of the current directory by using the notation dot (`.`). The dot (`.`) notation is commonly used when running programs that read the current directory name.

Creating directories (mkdir command)

Use the **mkdir** command to create one or more directories specified by the *Directory* parameter.

Each new directory contains the standard entries dot (`.`) and dot dot (`..`). You can specify the permissions for the new directories with the **-m** *Mode* flag.

When you create a directory, it is created within the current, or working, directory unless you specify an absolute path name to another location in the file system.

The following are examples of how to use the **mkdir** command:

- To create a new directory called Test in the current working directory with default permissions, type the following:

```
mkdir Test
```

- To create a directory called Test with `rxwxr-xr-x` permissions in a previously created `/home/demo/sub1` directory, type the following:

```
mkdir -m 755 /home/demo/sub1/Test
```

- To create a directory called Test with default permissions in the `/home/demo/sub2` directory, type the following:

```
mkdir -p /home/demo/sub2/Test
```

The **-p** flag creates the `/home`, `/home/demo`, and `/home/demo/sub2` directories if they do not already exist.

Moving or renaming directories (mvdir command)

Use the **mvdir** command to move or rename a directory.

The following are examples of how to use the **mvdir** command:

- To move a directory, type the following:

```
mvmkdir book manual
```

This moves the book directory under the directory named manual, if the manual directory exists. Otherwise, the book directory is renamed to manual.

- To move and rename a directory, type the following:

```
mvmkdir book3 proj4/manual
```

If a directory named manual already exists, this moves book3 and its contents to proj4/manual. In other words, book3 becomes a subdirectory of proj4/manual. If manual does not exist, this renames the book3 directory to proj4/manual.

Displaying the current directory (pwd command)

Use the **pwd** command to write to standard output the full path name of your current directory (from the / (root) directory).

All directories are separated by a slash (/). The / (root) directory is represented by the first slash (/), and the last directory named is your current directory.

For example, to display your current directory, type the following:

```
pwd
```

The full path name of your current directory displays similar to the following:

```
/home/thomas
```

Changing to another directory (cd command)

Use the **cd** command to move from your present directory to another directory. You must have execute (search) permission in the specified directory.

If you do not specify a *Directory* parameter, the **cd** command moves you to your login directory (\$HOME in the **ksh** and **bsh** environments, or \$home in the **cs**h environment). If the specified directory name is a full path name, it becomes the current directory. A full path name begins with a slash (/) indicating the / (root) directory, a dot (.) indicating current directory, or a dot dot (..) indicating parent directory. If the directory name is not a full path name, the **cd** command searches for it relative to one of the paths specified by the \$CDPATH shell variable (or \$cdpath **cs**h variable). This variable has the same syntax as, and similar semantics to, the \$PATH shell variable (or \$path **cs**h variable).

The following are examples of how to use the **cd** command:

- To change to your home directory, type the following:

```
cd
```

- To change to the /usr/include directory, type the following:

```
cd /usr/include
```

- To go down one level of the directory tree to the sys directory, type the following:

```
cd sys
```

If the current directory is /usr/include and it contains a subdirectory named sys, then /usr/include/sys becomes the current directory.

- To go up one level of the directory tree, type the following:

```
cd ..
```

The special file name, dot dot (..), refers to the directory immediately above the current directory, its parent directory.

Copying directories (cp command)

Use the **cp** command to create a copy of the contents of the file or directory specified by the *SourceFile* or *SourceDirectory* parameters into the file or directory specified by the *TargetFile* or *TargetDirectory* parameters.

If the file specified as the *TargetFile* exists, the copy writes over the original contents of the file. If you are copying more than one *SourceFile*, the target must be a directory.

To place a copy of the *SourceFile* into a directory, specify a path to an existing directory for the *TargetDirectory* parameter. Files maintain their respective names when copied to a directory unless you specify a new file name at the end of the path. The **cp** command also copies entire directories into other directories if you specify the **-r** or **-R** flags.

The following are examples of how to use the **cp** command:

- To copy all the files in the `/home/accounts/customers/orders` directory to the `/home/accounts/customers/shipments` directory, type the following:

```
cp /home/accounts/customers/orders/* /home/accounts/customers/shipments
```

This copies the files, but not the directories, from the `orders` directory into the `shipments` directory.

- To copy a directory, including all its files and subdirectories, to another directory, type the following:

```
cp -R /home/accounts/customers /home/accounts/vendors
```

This copies the `customers` directory, including all its files, subdirectories, and the files in those subdirectories, into the `vendors` directory.

For more information on **cp** command with all the available flags, see [cp command](#).

Displaying contents of a directory

Use the **ls** command to display the contents of a directory.

The **ls** command writes to standard output the contents of each specified *Directory* or the name of each specified *File*, along with any other information you ask for with the flags. If you do not specify a *File* or *Directory*, the **ls** command displays the contents of the current directory.

By default, the **ls** command displays all information in alphabetic order by file name. If the command is executed by a user with root authority, it uses the **-A** flag by default, listing all entries except dot (`.`) and dot dot (`..`). To show all entries for files, including those that begin with a dot (`.`), use the **ls -a** command.

You can format the output in the following ways:

- List one entry per line, using the **-l** flag.
- List entries in multiple columns, by specifying either the **-C** or **-x** flag. The **-C** flag is the default format when output is to a tty.
- List entries in a comma-separated series by specifying the **-m** flag.

To determine the number of character positions in the output line, the **ls** command uses the `$COLUMNS` environment variable. If this variable is not set, the command reads the `terminfo` file. If the **ls** command cannot determine the number of character positions by either of these methods, it uses a default value of 80.

The information displayed with the **-e** and **-l** flags is interpreted as follows:

The first character of each entry may be one of the following:

Ite	Description
-----	-------------

m	
---	--

d	Entry is a directory.
---	-----------------------

Ite	Description
------------	--------------------

- | | |
|----------|--|
| m | |
| b | Entry is a block special file. |
| c | Entry is a character special file. |
| l | Entry is a symbolic link. |
| p | Entry is a first-in, first-out (FIFO) pipe special file. |
| s | Entry is a local socket. |
| - | Entry is an ordinary file. |

The next nine characters are divided into three sets of three characters each. The first three characters show the file or directory owner's permission. The next three characters show the permission of the other users in the group. The last three characters show the permission of anyone else with access to the file. The three characters in each set show read, write, and execute permission of the file. Execute permission of a directory lets you search a directory for a specified file.

Permissions are indicated as follows:

Ite	Description
------------	--------------------

- | | |
|----------|--|
| m | |
| r | Read permission granted |
| t | Only the directory owner or the file owner can delete or rename a file within that directory, even if others have write permission to the directory. |
| w | Write (edit) permission granted |
| x | Execute (search) permission granted |
| - | Corresponding permission not granted. |

The information displayed with the **-e** flag is the same as with the **-l** flag, except for the addition of an 11th character, interpreted as follows:

Ite	Description
------------	--------------------

- | | |
|----------|---|
| m | |
| + | Indicates a file has extended security information. For example, the file might have extended ACL, TCB, or TP attributes in the mode. |
| - | Indicates a file does not have extended security information. |

When the size of the files in a directory are listed, the **ls** command displays a total count of blocks, including indirect blocks.

See the following examples:

- To list all files in the current directory, type the following:

```
ls -a
```

This lists all files, including

- dot (.)
 - dot dot (..)
 - Other files whose names might or might not begin with a dot (.)
- To display detailed information, type the following:

```
ls -l chap1 .profile
```

This displays a long listing with detailed information about `chap1` and `.profile`.

- To display detailed information about a directory, type the following:

```
ls -d -l . manual manual/chap1
```

This displays a long listing for the directories `.` and `manual`, and for the file `manual/chap1`. Without the **-d** flag, this would list the files in the `.` and `manual` directories instead of the detailed information about the directories themselves.

Deleting or removing directories (**rmdir** command)

Use the **rmdir** command to remove the directory, specified by the *Directory* parameter, from the system.

The directory must be empty (it can contain only `.` and `..`) before you can remove it, and you must have write permission in its parent directory. Use the **ls -a***Directory* command to check whether the directory is empty.

The following are examples of how to use the **rmdir** command:

- To empty and remove a directory, type the following:

```
rm mydir/* mydir/.  
rmdir mydir
```

This removes the contents of `mydir`, then removes the empty directory. The **rm** command displays an error message about trying to remove the directories `.` and `..`, and then the **rmdir** command removes them and the directory itself.

Note: **rm mydir/* mydir/.*** first removes files with names that do not begin with a dot, and then removes those with names that do begin with a dot. The **ls** command does not list file names that begin with a dot unless you use the **-a** flag.

- To remove the `/tmp/jones/demo/mydir` directory and all the directories beneath it, type the following:

```
cd /tmp  
rmdir -p jones/demo/mydir
```

This removes the `jones/demo/mydir` directory from the `/tmp` directory. If a directory is not empty or you do not have write permission to it when it is to be removed, the command terminates with appropriate error messages.

Comparing the contents of directories (**dircmp** command)

Use the **dircmp** command to compare two directories specified by the *Directory1* and *Directory2* parameters and write information about their contents to standard output.

First, the **dircmp** command compares the file names in each directory. If the same file name is contained in both, the **dircmp** command compares the contents of both files.

In the output, the **dircmp** command lists the files unique to each directory. It then lists the files with identical names in both directories, but with different contents. If no flag is specified, it also lists files that have identical contents as well as identical names in both directories.

The following are examples of how to use the **dircmp** command:

- To summarize the differences between the files in the `proj.ver1` and `proj.ver2` directories, type the following:

```
dircmp proj.ver1 proj.ver2
```

This displays a summary of the differences between the `proj.ver1` and `proj.ver2` directories. The summary lists separately the files found only in one directory or the other, and those found in both. If a file is found in both directories, the **dircmp** command notes whether the two copies are identical.

- To show the details of the differences between the files in the `proj.ver1` and `proj.ver2` directories, type the following:

```
dircmp -d -s proj.ver1 proj.ver2
```

The **-s** flag suppresses information about identical files. The **-d** flag displays a **diff** listing for each of the differing files found in both directories.

Command summary for file systems and directories

The following are commands for file systems and directories, commands for directory-handling procedures, and a list of directory abbreviations.

Table 7. Command summary for file systems

Item	Description
df	Reports information about space on file systems.

Table 8. Directory abbreviations

Item	Description
.	The current working directory.
..	The directory above the current working directory (the parent directory).
~	Your home directory. (This is not true for the Bourne shell. For more information, see Bourne Shell .)
\$HOME	Your home directory. (This is true for all shells.)

Table 9. Command summary for directory-handling procedures

Item	Description
cd	Changes the current directory.
cp	Copies files or directories.
dircmp	Compares two directories and the contents of their common files.
ls	Displays the contents of a directory.
mkdir	Creates one or more new directories.
mvdir	Moves (renames) a directory.
pwd	Displays the path name of the working directory.
rmdir	Removes a directory.

Workload manager

Workload Manager (WLM) is designed to provide the system administrator with increased control over how the scheduler virtual memory manager (VMM) and the disk I/O subsystem allocate resources to processes.

You can use WLM to prevent different classes of jobs from interfering with each other and to allocate resources based on the requirements of different groups of users.



Attention: Efficient use of WLM requires extensive knowledge of existing system processes and performance. If the system administrator configures WLM with extreme or inaccurate values, performance will be significantly degraded.

WLM is primarily intended for use with large systems. Large systems are often used for server consolidation, in which workloads from many different server systems (such as printer, database, general

user, and transaction processing systems) are combined into a single large system to reduce the cost of system maintenance. These workloads often interfere with each other and have different goals and service agreements.

WLM also provides isolation between user communities with very different system behaviors. This can prevent effective starvation of workloads with certain behaviors (for example, interactive or low CPU usage jobs) by workloads with other behaviors (for example, batch or high memory usage jobs).

Also, WLM ties into the accounting subsystem allowing users to do resource usage accounting per WLM class in addition to the standard accounting per user or group.

Workload management concepts

With WLM, you can create different classes of service for jobs, as well as specify attributes for those classes.

These attributes specify minimum and maximum amounts of CPU, physical memory, and disk I/O throughput to be allocated to a class. WLM then assigns jobs automatically to classes using class assignment rules provided by a system administrator. These assignment rules are based on the values of a set of attributes for a process. Either the system administrator or a privileged user can also manually assign jobs to classes, overriding the automatic assignment.

Terminology for workload management

Common terms associated with workload management are listed and described in this table.

Item	Description
class	A <i>class</i> is a collection of processes and their associated threads. A class has a single set of resource-limitation values and target shares. <i>class</i> is used to describe both subclasses and super classes.
superclass	A <i>superclass</i> is a class that has subclasses associated with it. No processes can belong to a superclass without also belonging to a subclass. A super class has a set of class-assignment rules that determines which processes are assigned to the superclass. A super class also has a set of resource-limitation values and resource target shares that determines the amount of resources that can be used by processes which belong to the superclass. These resources are divided among the subclasses based on the resources limitation values and resource target shares of the subclasses.
subclasses	<p>A <i>subclass</i> is a class associated with exactly one superclass. Every process in a subclass is also a member of its superclass. Subclasses have access only to resources that are available to the superclass. A subclass has a set of class assignment rules that determines which of the processes assigned to the superclass belong to the subclass. A subclass also has a set of resource-limitation values and resource target shares that determines the resources that can be used by processes in the subclass.</p> <p>These resource-limitation values and resource target shares indicate how much of the resources available to the superclass (the target for the superclass) can be used by processes in the subclass.</p> <p>WLM administration can be done using SMIT, or the WLM command-line interface.</p>
classification mechanism	A <i>classification mechanism</i> is a set of class assignment rules that determines which processes are assigned to which classes (super classes or subclasses within super classes).

Item	Description
class assignment rule	A <i>class assignment rule</i> indicates which values within a set of process attributes result in a process being assigned to a particular class (superclass or subclass within a superclass).
process attribute value	A <i>process attribute value</i> is the value that a process has for a process attribute. The process attributes can include attributes such as user ID, group ID, and application path name.
resource-limitation values	<i>Resource-limitation values</i> are a set of values that WLM maintains for a set of resource utilization values. These limits are completely independent of the resource limits specified with the setrlimit subroutine.
resource target share	<i>Resource target shares</i> are the shares of a resource that are available to a class (subclass or superclass). These shares are used with other class shares (subclass or superclass) at the same level and tier to determine the desired distribution of the resources between classes at that level and tier.
resource-utilization value	A <i>resource-utilization value</i> is the amount of a resource that a process or set of processes is currently using in a system. Whether it is one process or a set of processes is determined by the scope of process resource collection.
scope-of-resource collection	The <i>scope-of-resource collection</i> is the level at which resource utilization is collected and the level at which resource-limitation values are applied. This might be at the level of each process in a class, the level of the sum across every process in a class owned by each user, or the level of the sum across every process in a class. The only scope currently supported is the latter.
process class properties	The <i>process class properties</i> are the set of properties that are given to a process based on the classes (subclass and superclass) to which it is assigned.
class authorizations	The <i>class authorizations</i> are a set of rules that indicates which users and groups are allowed to perform operations on a class or processes and threads in a class. This includes the authorization to manually assign processes to a class or to create subclasses of a superclass.
class tier	The <i>class tier</i> value is the position of the class within the hierarchy of resource limitation desirability for all classes. The resource limits (including the resource targets) for all classes in a tier are satisfied before any resource is provided to lower tier classes. Tiers are provided at both the superclass and subclass levels. Resources are provided to super classes based on their tiers. Within a superclass, resources are given to subclasses based on their tier values within the superclass. Thus, superclass tier is the major differentiator in resource distribution; the subclass tier provides an additional smaller differentiator within a superclass.

Classes for workload management

WLM allows system administrators to define classes and define for each class a set of attributes and resource limits.

The processes are assigned to classes based on criteria provided by the system administrator. The resource entitlements and limits are enforced at the class level. This method of defining classes of service and regulating the resource utilization of each class of applications prevents applications with very different resource use patterns from interfering with each other when they share a single server.

WLM supports a hierarchy of classes with two levels:

- The resources of the system are distributed among superclasses according to the resource entitlements for each superclass. The system administrator defines resource entitlements.
- In turn, each superclass can have subclasses. The resources allocated to the superclass are distributed among the subclasses according to the resource entitlements given to each subclass.
- The system administrator can delegate the administration of the subclasses of each superclass to a *superclass administrator* or to a group of superclass administrators.
- WLM supports up to 69 superclasses (64 user-defined) and 64 subclasses per superclass (61 user-defined).
- Depending on the needs of the organization, a system administrator can decide to use only superclasses or to use superclasses and subclasses.

Note: Throughout this discussion of WLM, the term *class* applies to both superclasses and subclasses. If discussion applies only to a specific class type, that type is explicitly mentioned.

Process assignment to classes for workload management

The processes are assigned to a class, using class-assignment rules provided by the system administrator. The classification criteria are based on the value of a set of attributes of the process such as user ID, group ID, name of the application file, type of process, and application tag.

A defined set of rules is used to determine the superclass a process is assigned to. If this superclass has subclasses defined, there is another set of rules for this superclass to determine which subclass is assigned to which process. This automatic assignment process also takes into account the inheritance attributes of both the superclass and the subclass.

The automatic class assignment is done when a process calls the **exec** subroutine. The class assignment is reevaluated when a process uses a subroutine that can alter a process attribute used for classification purposes. Examples are the **setuid**, **setgid**, **setpri**, and **plock** subroutines.

In addition to this automatic class assignment, a user with the proper authority can manually assign processes or groups of processes to a specific superclass or subclass.

Related concepts

Class attributes

List all the attributes of a WLM class.

Resource control

WLM allows management of resources in two ways: as a percentage of available resources or as total resource usage.

Resources that can be controlled on a percentage basis include the following:

- Processor use of the threads of type SCHED_OTHER in a class. This is the sum of all of the processor cycles consumed by every thread in the class. Fixed-priority threads are non-adjustable. Therefore, they cannot be altered, and they can exceed the processor usage target.
- Physical memory utilization of the processes in a class. This is the sum of all the memory pages that belong to the processes in the class.
- Disk I/O bandwidth of the class. This is the bandwidth (in 512-byte blocks per second) of all the I/Os started by threads in the class on each disk device accessed by the class.

Resources that can be controlled on a total usage basis fall into one of two categories: class totals or process totals. The class totals category includes:

Number of processes in a class

This is the number of processes that are active in a class at one time.

Number of threads in a class

This is the number of threads that are active in a class at one time.

Number of logins in a class

This is the number of login sessions that are active in a class at one time.

The process totals category includes:

Total CPU time

This is the total accumulated CPU time for a single process.

Total disk I/O

This is the total accumulated blocks of disk I/O for a single process.

Total connect time

This is total amount of time that a login session can be active.

Resource entitlements

WLM allows system administrators to specify per-class resource entitlements independently for each resource type.

These entitlements can be specified by indicating the following:

- The target for usage of different types of resources. This target is specified with shares. The shares are specified as relative amounts of usage between different classes. For instance, if two classes have respectively 1 and 3 shares of CPU and are the only classes active at this time, their percentage goal used by WLM for its CPU regulation will be 25% and 75%, respectively. The target percentages are calculated for classes in each tier based on the number of active shares in the tier and the amount of resource available to the tier.
- Minimum and maximum limits. These limits are specified as percentages of the total resource available. WLM supports two kinds of maximum limits:
 - A soft maximum limit indicates the maximum amount of the resource that can be made available when there is contention for the resource. This maximum can be exceeded if there is no contention; that is, if no one else requires the resource.
 - A hard maximum limit indicates the maximum amount of the resource that can be made available regardless of whether there is contention on the resource. Fixed-priority threads, however, are not subject to these same rules and therefore can exceed the limit.
- Total limits. The total limits are strictly enforced. If a process exceeds one of its total consumption limits, it will be terminated. If a class is at one of its total limits, any operation that would result in the creation of another instance of that resource will fail.

In most cases, soft maximum limits are sufficient to ensure that resource entitlements are met and enforced. Using hard maximum limits may result in unused system resources since these are strictly enforced, even when there is no contention for the resource. Careful consideration must be made when using hard maximum limits since these can greatly affect system or application performance if set too low. Total limits should also be used with caution, since these could result in process termination or failure to function as intended.

In active mode, WLM attempts to keep active classes close to their targets. Since there are few constraints on the values of the various limits, the sum of any of the limits across all classes could far exceed 100%. In this case, if all of the classes are active, the limit cannot be reached by all classes. WLM regulates the processor consumption by adjusting the scheduling priorities of the non-fixed priority threads in the system according to how the class they belong to is performing, relative to its limits and target. This approach guarantees a processor consumption averaged over a given period of time, rather than the processor consumption over very short intervals (for example, 10 ms).

For example, if class A is the only active class, with a processor minimum of 0% and a processor target of 60 shares, then it gets 100% of the processor. If class B, with a processor minimum limit of 0% and a processor target of 40 shares, becomes active, then the class A processor utilization progressively decreases to 60% and the class B processor utilization increases from 0% to 40%. The system stabilizes at 60% and 40% processor utilization, respectively, in a matter of seconds.

This example supposes that there is no memory contention between the classes. Under regular working conditions, the limits you set for processor and memory are interdependent. For example, a class may be unable to reach its target or even its minimum processor allocation if the maximum limit on its memory usage is too low compared to its working set.

To help refine the class definition and class limits for a given set of applications, WLM provides the **wlmstat** reporting tool, which shows the amount of resource currently being used by each class. A graphical display tool, **wlmon**, is also provided for system monitoring.

Workload Manager virtual memory limits

Workload Manager (WLM) virtual memory limits provide administrators a means to prevent system degradation or system failure due to excessive paging by providing a virtual memory limit on a class or a process.

When a limit is exceeded, WLM takes action by doing one of the following:

- killing all processes under the WLM class that exceeded its limit
- killing only the process that caused the WLM class usage to exceed its limit
- killing the process that exceeded its process limit

Virtual memory limits can be specified for any user-defined class, any default subclass under a user-defined super class, and the default super class.

For accounting purposes, WLM will only consider the following as virtual memory when determining WLM total class or process usage:

- heap
- loader initialized data, BSS, shared library, and privately loaded segments
- UBLOCK and mmap areas
- large and pinned user space pages

An administrator can specify a WLM virtual memory limit for a class or for each process in the class. When a class limit is exceeded, WLM can either kill all processes assigned to the class, or only kill the process that caused the limit to be exceeded, depending on whether the **vmenforce** class attribute is set to **class** or **proc**, respectively. The default behavior is to only kill the process that caused the limit to be exceeded. A process limit is killed if the virtual memory use of the process surpasses the limit.

Modes of operation for Workload Manager

WLM can be used to regulate resource consumption as per-class percentages, per-class totals, or per-process totals. Regulation for all resource types can be enabled by running WLM in active mode.

Optionally, you can start a mode of WLM that classifies new and existing processes and monitors the resource usage of the various classes, without attempting to regulate this usage. This mode is called the *passive mode*.

The passive mode can be used when configuring WLM on a new system to verify the classification and assignment rules, and to establish a base line of resource utilization for the various classes when WLM does *not* regulate the processor and memory allocation. This should give a basis for system administrators to decide how to apply the resource shares and resource limits (if needed) to favor critical applications and restrict less important work in order to meet their business goals.

If processor time is the only resource that you are interested in regulating, you can choose to run WLM in active mode for processor and passive mode for all other resources. This mode is called *cpu only* mode. If you want to regulate per-class percentages, but neither of the total resource types, the total resource accounting and regulation can be disabled for per-class totals, per-process totals, or both. In all modes, you have the option of disabling resource set binding.

Dynamic control of Workload Manager

When WLM is active, any parameter of the current configuration can be modified at any time, including the attributes of a class, its resource shares and limits, the assignment rules, and adding new classes or deleting existing classes.

This can be done in several ways, such as:

- Modifying the property files for the currently active configuration (directory pointed to by the symbolic link `/etc/wlm/current`) and refreshing WLM by using the **wlmcntrl** command to use the new parameters.
- Creating another configuration with a different set of parameters and updating WLM to load the parameters of the new configuration, thus making it the current configuration.
- Modifying some of the parameters of the currently active configuration using the WLM command line interface (the **mkclass**, **chclass**, and **rmclass** commands).
- Modifying some of the parameters of the currently active configuration from an application using the WLM APIs.

Automatic switches to a new configuration at specified times of day can be accomplished using *configuration sets*. Configuration sets allow the administrator to specify a set of configurations to be used, and a time range for which each will be active.

Monitoring tools

Use these WLM commands to display WLM statistics and monitor the operation of WLM.

- The **wlmstat** command is text oriented and displays statistics as text (percentage of resource utilization per class for all the resource types managed by WLM).
- The **wlmon** command gives a graphical view of per-class resource utilization and WLM regulation.
- The **wlmpperf** command is an optional tool available with the Performance Toolbox and provides more capabilities, such as long-term record and replay.

Backward compatibility in Workload Manager

The default output of the **wlmstat** command lists only the superclasses and is similar to those of previous versions.

For example:

```
# wlmstat
      CLASS  CPU  MEM  DKIO
Unclassified  0    0    0
Unmanaged    0    0    0
Default      0    0    0
Shared       0    2    0
System       2   12    0
class1       0    0    0
class2       0    0    0
#
```

If some of the superclasses have subclasses defined by a WLM administrator, then the subclasses are listed. For example:

```
# wlmstat
      CLASS  CPU  MEM  DKIO
Unclassified  0    0    0
Unmanaged    0    0    0
Default      0    0    0
Shared       0    2    0
System       3   11    7
class1      46    0    0
class1.Default 28    0    0
class1.Shared  0    0    0
class1.sub1   18    0    0
class2      48    0    0
#
```

The output is the same when you use the **ps** command. For processes in a superclass without any subclasses, the **ps** command lists the superclass name as the process class name.

Per class accounting

The AIX accounting system utility lets you collect and report the use of various system resources by user, group, or WLM class.

When process accounting is turned on, the operating system records statistics about the process resource usage in an accounting file when the process exits. This accounting record includes a 64-bit numeric key representing the name of the WLM class that the process belonged to.

The accounting subsystem uses a 64-bit key instead of the full 34-character class name to save space (otherwise the change would practically double the size of the accounting record). When the accounting command is run to extract the per-process data, the key is translated back into a class name using the above-mentioned routine. This translation uses the class names currently in the WLM configuration files. So, if a class has been deleted between the time the accounting record was written, when the process terminated, and the time the accounting report is run, the class name corresponding to the key cannot be found, and the class displays as Unknown.

To keep accurate records of the resource usage of classes deleted during an accounting period, do one of the following:

- Instead of deleting the class, keep the class name in the classes file and remove the class from the rules file so that no process can be assigned to it. Then you can delete the class after the accounting report has been generated at the end of the accounting period.
- Or, delete the class from the configuration it belongs to, and keep the class name in the classes file in a "dummy" configuration (one that is never activated) until after the accounting records for the period have been generated.

Administering Workload Manager

Workload Manager (WLM) gives system administrators more control over how the scheduler and the virtual memory manager (VMM) allocate resources to processes. Using WLM, you can prevent different classes of jobs from interfering with each other and you can allocate resources based on the requirements of different groups of users.

WLM lets you create different classes of service for jobs, as well as specify attributes for those classes. These attributes specify minimum and maximum amounts of CPU, physical memory, and disk I/O throughput to be allocated to a class. WLM then assigns jobs automatically to classes using class assignment rules provided by a system administrator. These assignment rules are based on the values of a set of attributes for a process. Either the system administrator or a privileged user can also manually assign jobs to classes, overriding the automatic assignment.

WLM is part of the base operating system and is installed with the base operating system, but it is an optional service. It must be configured to suit your system environment, started when you want to use it, and stopped when you want to suspend or end WLM service.

This section contains procedures for configuring WLM with classes and rules that are appropriate for your site and suggestions for troubleshooting unexpected resource consumption behavior.



Attention: The tasks in this section assume you are familiar with WLM concepts. Efficient use of WLM requires extensive knowledge of existing system processes and performance. If the system administrator configures WLM with extreme or inaccurate values, performance will be significantly degraded.

Starting up and shutting down Workload Manager

WLM is an optional service that must be started and stopped.

It is recommended that you use one of the system management interfaces, SMIT, to start or stop WLM.

- To start or stop WLM using SMIT, use the `smit wlmmanage` fast path.

The key difference between these options is permanence. In SMIT, you can start or stop WLM three ways:

current session

If you request to stop WLM with this option, WLM will be stopped for this session only and restarted at next reboot. If you request a start with this option, WLM will be started for this session only and not restarted at next reboot.

next reboot

If you request to stop WLM with this option, WLM will remain running for this session only and *will not be* restarted at next reboot. If you request a start with this option, WLM will not be available for this session, but will be started at next reboot.

both

If you request to stop WLM with this option, WLM will be stopped for this session only and *will not be* restarted at next reboot. If you request a start with this option, WLM will be started for this session only *and* will be restarted at next reboot.

You can also use the **wlmcntrl** command, but the **wlmcntrl** command allows you to start or stop WLM for the current session only. If you want to use the command line interface and you want the change to remain in effect when the machine is rebooted, you must edit the `/etc/inittab` file.

WLM can be used to regulate resource consumption as per-class percentages, per-class totals, or per-process totals. Regulation for all resource types can be enabled by running WLM in *active* mode. Optionally, you can start a mode of WLM that classifies new and existing processes and monitors the resource usage of the various classes, without attempting to regulate this usage. This mode is called the *passive* mode. If CPU time is the only resource that you are interested in regulating, you can choose to run WLM in active mode for CPU and passive mode for all other resources. This mode is called *cpu only* mode.

All processes existing in the system before WLM is started are classified according to the newly loaded assignment rules, and are monitored by WLM.

Workload Manager properties

You can specify the properties for the WLM configuration by using the SMIT, the WLM command line interface, or by creating flat ASCII files. The SMIT interfaces use the WLM commands to record the information in the same flat ASCII files, called *property files*.

A set of WLM property files defines a WLM configuration. You can create multiple sets of property files, defining different configurations of workload management. These configurations are located in subdirectories of `/etc/wlm`. The WLM property files describing the super classes of the *Config* configuration are the file's *classes*, *description*, *limits*, *shares* and *rules* in `/etc/wlm/Config`. Then, the property file's describing the subclasses of the superclass *Super* of this configuration are the file's *classes*, *limits*, *shares* and *rules* in directory `/etc/wlm/Config/Super`. Only the root user can start or stop WLM, or switch from one configuration to another.

The property files are named as follows:

Item	Description
classes	Class definitions
description	Configuration description text
groupings	Attribute value groupings
limits	Class limits
shares	Class target shares
rules	Class assignment rules

The command to submit the WLM property files, **wlmcntrl**, and the other WLM commands allow users to specify an alternate directory name for the WLM properties files. This allows you to change the WLM properties without altering the default WLM property files.

A symbolic link, `/etc/wlm/current`, points to the directory containing the current configuration files. Update this link with the **wlmcntrl** command when you start WLM with a specified configuration or

configuration set. The sample configuration files shipped with the operating system are in `/etc/wlm/` standard.

Creating an attribute value grouping

You can group attribute values and represent them with a single value in the rules file. These *attribute value groupings* are defined in a `groupings` file within the WLM configuration directory.

By default, a configuration has no `groupings` file. There is no command or management interface to create one. To create and use attribute value groupings, use the following procedure:

1. With root authority, change to the appropriate configuration directory, as shown in the following example:

```
cd /etc/wlm/MyConfig
```

2. Use your favorite editor to create and edit a file named `groupings`.
For example:

```
vi groupings
```

3. Define attributes and their associated values using the following format:

attribute = *value*, *value*, ...

All values must be separated by commas. Spaces are not significant. Ranges and wild cards are allowed. For example:

```
trusted = user[0-9][0-9], admin*
nottrusted = user23, user45
shell = /bin/?sh, \
        /bin/sh, \
        /bin/tcsh
rootgroup=system,bin,sys,security,cron,audit
```

4. Save the file.
5. To use attribute groupings within the selection criteria for a class, edit the `rules` file.

The attribute grouping name must be preceded by a dollar sign (\$) to include the corresponding values or the exclamation point (!) to exclude the values. The exclamation point cannot be used in the members of the group (step “3” on page 135), and it is the only modifier that can be used in front of the grouping in this rules file. In the following example, the asterisk (*) signals a comment line:

*class	resvd	user	group	application	type	tag
classA	-	\$trusted,!\$nottrusted	-	-	-	-
classB	-	-	-	\$shell,!/bin/zsh	-	-
classC	-	-	\$rootgroup	-	-	-

6. Save the file.

At this point, your classification rules includes attribute value groupings. When the rules are parsed, if an element begins with a \$, the system looks for that element within the `groupings` file. If an element is syntactically invalid or if the `groupings` file does not exist, the system displays a warning message and continues processing other rules.

Creating a time-based configuration set

You can create a set of specialty configurations and assign each configuration within the set to days and times when you want a specific configuration to be in effect.

These sets, called *time-based configuration sets*, are completely separate from but compatible with your normal configuration. You can use the **wlmcntrl -u** command to switch between a configuration set and your normal configuration as needed.

When using a configuration set, you associate existing named configurations, typically with a specific time range. Because only one configuration can be used at any given time, each specified time range must be unique; time ranges cannot overlap or be duplicated.

The **wlmd** daemon alerts WLM when a specified configuration goes out of time range and another configuration needs to be used. Only the root user can manage these time ranges, which are specified within the configuration set's directory in an ASCII file called `.times`.

Use the following procedure to create a time-based configuration set:

1. With root authority, create a configuration set directory then change to that directory.
For example:

```
mkdir /etc/wlm/MyConfigSet
cd /etc/wlm/MyConfigSet
```

2. Use your favorite editor to create the configuration set's `.times` file and specify the configuration and time ranges in the following format:

```
ConfigurationName:
    time = "N-N,HH:MM-HH:MM"
```

or

```
ConfigurationName:
    time = -
```

(no time value specified)

Where *N* is a numeral representing a day of the week in the range of 0 (Sunday) through 6 (Saturday), *HH* represents the hour in the range of 00 (midnight) to 23 (11 p.m.), and *MM* represents the minutes in the range of 00 to 59. You can specify the day only or not at all. An hour value of 24 is valid for the ending hour of the day, provided that the minute value is 00. If you type a dash (-) instead of a time range for a particular configuration, that configuration will be used when the other configurations' time ranges are not in effect. Only one configuration can be specified without a time range.

For example:

```
conf1:
    time =
conf2:
    time = "1-5,8:00-17:00"
conf2
    time = "6-0,14:00-17:00"
conf3
    time = "22:00-6:00"
```

3. Use the **wlmcntrl -u** command to update WLM with the new configuration set.
For example:

```
wlmcntrl -u /etc/wlm/MyConfigSet
```

At this point, WLM's current configuration is your new time-based configuration set.

You can also use the [**confsetcntrl**](#) and [**lswlmconf**](#) commands to create and manipulate configuration sets.
For example:

To create the `confset1` configuration set with a default configuration of `conf1`, use the following command:

```
confsetcntrl -C confset1 conf1
```

To add `conf2` to `confset1` and make it the active configuration from 8:00 AM to 5:00 PM daily, use the following command:

```
confsetcntrl -d confset1 -a conf2 "0-6,08:00-17:00"
```

To make this configuration set the active configuration, use the following command:

```
wlmcntrl -d confset1
```


Creating a resource set

Using resource sets (rsets) is an effective way to isolate workloads from one another as far as the CPU is concerned. By separating two different workloads into two classes and giving each class a different subset of the CPUs, you can make sure that the two workloads never compete with each other for CPU resources, even though they still compete for physical memory and I/O bandwidth.

The simplest way to create a resource set is to use the SMIT interface (**smit addrsetcntl** fast path) or the **mkrset** command.

For instructional purposes, the following example illustrates each step of creating and naming a resource set on a 4-way system. Its goal is to create a resource set containing processors 0 to 2, and use it in WLM configuration to restrict all processes of a superclass to these three processors.

1. With root authority, view the available building blocks (from which to create the resource sets) using the following command:

```
lsrset -av
```

The output for this example is the following:

T	Name	Owner	Group	Mode	CPU	Memory	Resources
r	sys/sys0	root	system	r-----	4	98298	sys/sys0
r	sys/node.00000	root	system	r-----	4	98298	sys/sys0
r	sys/mem.00000	root	system	r-----	0	98298	sys/mem.00000
r	sys/cpu.00003	root	system	r-----	1	0	sys/cpu.00003
r	sys/cpu.00002	root	system	r-----	1	0	sys/cpu.00002
r	sys/cpu.00001	root	system	r-----	1	0	sys/cpu.00001
r	sys/cpu.00000	root	system	r-----	1	0	sys/cpu.00000

In the output, **sys/sys0** represents the whole system (in this case, a 4-way SMP). When a WLM class does not specify an **rset** attribute, this is the default set that its processes potentially can access.

2. Create and name the resource set using the following SMIT fast path:

```
smit addrsetcntl
```

For this example, fill in the fields as follows:

Name Space

admin

Resource Set Name

proc0_2

Resources

Select from the list those lines that correspond to the memory and CPUs 0 to 2 (sys/cpu.00000 to sys.cpu.00002).

All other fields

Select from the lists.

When you finish entering the fields and exit SMIT, the **admin/proc0_2** rset is created in **/etc/rsets**.

3. To use the new rset, add it into the kernel data structures using the following SMIT fast path:

```
smit reloadrsetcntl
```

This menu gives you the option to reload the database now, at **next boot** or **both**. Because this is the first time you are using the new resource set, select **both** so that this rset will be loaded now and after each reboot. (If you had changed an existing rset, you would probably have selected **now**.)

4. Add the new rset to a WLM class using the following SMIT fast path:

```
smit wlmclass_gal
```

Select the class (in this example, **super1**) then select **admin/proc0_2** from the list available for the **Resource Set** field. After you make your selection and exit SMIT, the **classes** file on disk is changed.

5. Do one of the following:

- If WLM is running, update the configuration using the following SMIT fast path:

```
smit wlmupdate
```

- If WLM is not running, start it using the following SMIT fast path:

```
smit wlmstart
```

6. Monitor the effect of the new resource set on the class. For example:

- Start 90 CPU loops (program executing an infinite loop) in class **super1**.
- Type `wlmstat` on the command line. The output for this example is the following

	CLASS	CPU	MEM	BIO
	Unclassified	0	0	0
	Unmanaged	0	0	0
	Default	8	0	0
	Shared	0	0	0
	System	0	0	0
	super1	75	0	0
	super2	0	0	0
	super2.Default	0	0	0
	super2.Shared	0	0	0
	super2.sub1	0	0	0
	super2.sub2	0	0	0

This output shows that the 90 CPU bound processes, which otherwise unconstrained would take up 100% of the CPU, now use only 75% because the resource set limits them to run on CPUs 0 to 2.

- To verify what resource set a process (identified by its PID) has access to, use the following SMIT fast path:

```
smit lsrsetproc
```

Enter the PID of the process you are interested in or select it from the list. The following output is for one of the loop processes:

CPU	Memory	Resources
3	98298	sys/mem.000000 sys/cpu.000002 sys/cpu.000001 sys/cpu.000000

However, a process from a class without a specified `rset` attribute uses the `Default` resource set that includes all processors except those processors that are part of an exclusive resource set. A process that does not belong to any class uses the `System` class (if it is a root process) or a `Default` class (if it is a nonroot process). Either of these classes might have resource sets that are defined for them.

The following output is from the **init** process, which is in a class that does not specify a resource set:

CPU	Memory	Resources
4	98298	sys/sys0

At this point, your resource set exists and is being used by at least one class within WLM.

Note: WLM will not set its `rset` attachment for a process that currently has a **bindprocessor** subroutine binding or another `rset` attachment. When the other attachment no longer exists, WLM will assign its `rset` automatically.

Note: Resource sets can be created for any WLM class, which includes the `Default` and `System` classes.

Configuring Workload Manager to consolidate workloads

Workload Manager (WLM) gives you control over the resources used by jobs on your system.

A default WLM configuration template exists on every installed AIX operating system. The following procedure updates the WLM configuration template to implement a resource-management policy on a

shared server. The resulting configuration can be used as a starting point for testing. Exactly how you configure WLM will depend on the workload and policy requirements for your environment.

Note:

1. Efficient use of WLM requires extensive knowledge of existing system processes and performance. Repeated testing and tuning will probably be needed before you can develop a configuration that works well for your workload. If you configure WLM with extreme or inaccurate values, you can significantly degrade system performance.
2. The process of configuring WLM is simpler when you already know one or more of the classification attributes of a process (for example, user, group, or application name). If you are unfamiliar with the current use of resources, use a tool such as **topas** to identify the processes that are the primary resource users and use the resulting information as the starting point for defining classes and rules.
3. The following scenario assumes you are familiar with the basic Workload Manager concepts as described in [“Workload management concepts” on page 127](#).

The WLM configuration files exist in the `/etc/wlm/ConfigurationName` directory. Each subclass for each superclass is defined in a configuration file named `/etc/wlm/ConfigurationName/SuperClassName`. For more information about these files, see the *Files Reference*.

In the following procedure, you consolidate the workloads from two separate department servers onto one larger server. This example edits the configuration files, but you can also create a configuration using SMIT (use the **smitt wlmconfig_create** fast path). Briefly, in this procedure, you will do the following:

1. Identify the resource requirements of the applications you want to consolidate. This will help you determine how many applications you can move to the larger server.
2. Define tiers, as well as resource shares and limits, to begin testing with the consolidated workload.
3. Fine-tune the configuration until you achieve your desired results.

The information in this how-to scenario was tested using specific versions of AIX. The results you obtain might vary significantly depending on your version and level of AIX.

Step 1. Identify application requirements

In this scenario, the workload is typical of what you might see on a database server. Assume the jobs fall into the following general categories:

Listeners

These are processes that sleep most of the time and wake up periodically in response to a request. Although these processes do not consume a lot of resources, response time can be critical.

Workers

These are processes that do the work on behalf of a request, whether the request is local or remote. These processes probably use a lot of CPU time and memory.

Reporters

These are processes that do automated tasks. They might require a lot of CPU time or memory, but they can tolerate a slower response time.

Monitors

These are processes that typically run periodically to verify the state of the system or applications. These processes might use a significant amount of resource, but only for a short time.

Commands

These are commands or other applications that system users might run at any time. Their resource requirements are unpredictable.

In addition to this work, scheduled jobs fall into one of the following categories:

SysTools

These are processes that perform automated tasks. These jobs are not critical to system operation but need to run periodically and within certain time constraints.

SysBatch

These are processes that run infrequently, are not critical to system operation, and need not finish in a timely manner.

The first step of creating a configuration is to define classes and rules. In the following steps, you will use the general job categories listed above to define your classes. Use the following procedure:

1. Make a new configuration within the `/etc/wlm` directory called `MyConfig` using the following command:

```
mkdir /etc/wlm/MyConfig
```

2. Copy the template files into the `/etc/wlm/MyConfig` directory using the following command:

```
cp -pr /etc/wlm/template/* /etc/wlm/MyConfig
```

3. To create the super classes, use your favorite editor to modify the `/etc/wlm/MyConfig/classes` file to contain the following:

```
System:
Default:
DeptA:
DeptB:
SysTools:
SysBatch:
```

As a starting point, you define one superclass for each department (because two departments will be sharing the server). The `SysTool` and `SysBatch` super classes will handle the scheduled jobs outlined in the general categories above. The `System` and `Default` super classes are always defined.

4. Within the `MyConfig` directory, create a directory for each the `DeptA` and `DeptB` super classes. (When creating a configuration, you must create a directory for every superclass that has subclasses.) In the following step, you define five subclasses (one for each category of work) for each department's superclass.
5. To create subclasses for each general category of jobs, edit the `/etc/wlm/MyConfig/DeptA/classes` and `/etc/wlm/MyConfig/DeptB/classes` files to contain the following:

```
Listen:
Work:
Monitor:
Report:
Command:
```

Note: The contents of the `classes` file can be different for each superclass.

After the classes are identified, in the following step, you create the classification rules that are used to classify processes at the superclass and subclass levels. For the sake of simplicity, assume that all applications run from known locations, that all processes from one department run under the `deptA` UNIX group, and that processes from the other department run under the `deptB` UNIX group.

6. To create the superclass assignment rules, modify the `/etc/wlm/MyConfig/rules` file to contain the following:

```
DeptA - - deptA - -
DeptB - - deptB - -
SysTools - root,bin - /usr/sbin/tools/* -
SysBatch - root,bin - /usr/sbin/batch/* -
System - root - - -
Default - - - -
```

Note: If more than one instance of the same application can be running and all classification attributes (other than the tag) are the same, use the **wlmassign** command or **wlm_set_tag** subroutine to differentiate between them by assigning them to different classes.

7. To create more specific subclass rules, create the `/etc/wlm/MyConfig/DeptA/rules` and `/etc/wlm/MyConfig/DeptB/rules` files with the following content:

```
Listen - - - /opt/myapp/bin/listen* -
Work - - - /opt/myapp/bin/work* -
Monitor - - - /opt/bin/myapp/bin/monitor -
Report - - - /opt/bin/myapp/report* -
Command - - - /opt/commands/* -
```

8. To determine the resource-consumption behavior of each class, start WLM in passive mode using the following command:

```
wlmcntrl -p -d MyConfig
```

After starting WLM in passive mode, you can run each application separately at first to gain a finer perspective of its resource requirements. You can then run all applications simultaneously to better determine the interaction among all classes.

An alternative method of identifying the application resource requirements might be to first run WLM in passive mode on the separate servers from which you are consolidating applications. The disadvantages to this approach are that you would have to re-create the configurations on the larger system, and the required percentage of resources will likely be different on the larger system.

Step 2. Define Tiers, Shares, and Limits

A WLM configuration is an implementation of a resource-management policy. Running WLM in passive mode provides information that helps you determine whether your resource-management policy is reasonable for the given workload. You can now define tiers, shares, and limits to regulate your workload based on your resource-management policy.

For this scenario, assume you have the following requirements:

- The System class must have the highest priority and be guaranteed access to a percentage of system resources at all times.
- The SysTools class must have access to a certain percentage of resources at all times, but not so much that it will significantly impact the applications that are running in DeptA and DeptB.
- The SysBatch class cannot interfere with any of the other work on the system.
- DeptA will receive 60% of the available resources (meaning resources that are available to the classes with shares) and DeptB will receive 40%. Within DeptA and DeptB:
 - Processes in the Listen class must respond to requests with a low latency, but must not consume a lot of resources.
 - The Work class must be allowed to consume the most resources. The Monitor and Command classes must consume some resource, but less than the Work class.
 - The Report class cannot interfere with any of the other work.

In the following procedure, you define tiers, shares, and limits:

1. To create the superclass tiers, use your favorite editor to modify the `/etc/wlm/MyConfig/classes` file to contain the following:

```
System:
Default:
DeptA:
    localshm = yes
    adminuser = adminA
    authuser = adminA
    inheritance = yes
```

```

DeptB:
    localshm = yes
    adminuser = adminB
    authuser = adminB
    inheritance = yes

SysTools:
    localshm = yes

SysBatch:
    tier = 1
    localshm = yes

```

The SysBatch superclass is put in tier 1 because this class contains very low-priority jobs that you do not want to interfere with the rest of the work on the system. (When a tier is not specified, the class defaults to tier 0.) Administration of each department's superclass is defined by the adminuser and authuser attributes. The inheritance attribute is enabled for DeptA and DeptB. All new processes started in a class with inheritance will remain classified in that class.

2. To create subclass tiers for each group of jobs, modify the `/etc/wlm/MyConfig/DeptA/classes` and `/etc/wlm/MyConfig/DeptB/classes` files to contain the following:

```

Listen:

Work:

Monitor:

Report:
    tier = 1
Command:

```

3. To assign the initial shares for the superclasses, edit the `/etc/wlm/MyConfig/shares` file to contain the following:

```

DeptA:
    CPU = 3
    memory = 3

DeptB:
    CPU = 2
    memory = 2

```

Because you assigned a CPU total of 5 shares, DeptA processes will have access to three out of five shares (or 60%) of the total CPU resources and DeptB processes will have access to two out of five (or 40%). Because you did not assign shares to the SysTools, System, and Default classes, their consumption targets will remain independent from the number of active shares, which gives them higher-priority access to resources than the DeptA and DeptB (until their limit is reached). You did not assign the SysBatch class any shares because it is the only superclass in tier 1, and therefore any share assignment is irrelevant. Jobs in the SysBatch class can only consume resources that are unused by all classes in tier 0.

4. To assign the initial shares for the subclasses, edit the `/etc/wlm/MyConfig/DeptA/shares` and `/etc/wlm/MyConfig/DeptB/shares` files to contain the following:

```

Work:
    CPU = 5
    memory = 5

Monitor:
    CPU = 4
    memory = 1

Command:
    CPU = 1
    memory = 1

```

Because you did not assign shares to the Listen class, it will have the highest-priority access (in the superclass) to resources when it requires them. You assigned the largest number of shares to the Work class because it has the greatest resource requirements. Accordingly, you assigned shares to the

Monitor and Command classes based on their observed behavior and relative importance. You did not assign shares to the Report class because it is the only subclass in tier 1, and therefore any share assignment is irrelevant. Jobs in the Report class can only consume resources that are unused by subclasses in tier 0.

In the following step of this example, you assign limits to classes that were not assigned shares. (You can also assign limits to classes with shares. See [Managing Resources with WLM](#) for more information.)

5. To assign limits to the superclasses, edit the `/etc/wlm/MyConfig/limits` file to contain the following:

```
Default:
    CPU = 0%-10%;100%
    memory = 0%-10%;100%

SysTools:
    CPU = 0%-10%;100%
    memory = 0%-5%;100%

System:
    CPU = 5%-50%;100%
    memory = 5%-50%;100%
```

You assigned soft maximum limits to the System, SysTools, and Default classes to prevent them from significantly interfering with other work on the system. You assigned minimum limits to the System class for CPU and memory because this class contains processes that are essential to system operation, and it must be able to consume a guaranteed amount of resource.

6. To assign limits to the subclasses, edit the `/etc/wlm/MyConfig/DeptA/limits` and `/etc/wlm/MyConfig/DeptB/limits` files to contain the following:

```
Listen:
    CPU = 10%-30%;100%
    memory = 10%-20%;100%

Monitor:
    CPU = 0%-30%;100%
    memory = 0%-30%;100%
```

Note: The limits can be different for each subclass file.

You assigned soft maximum limits to the Listen and Monitor classes to prevent them from significantly interfering with the other subclasses in the same superclass. In particular, you do not want the system to continue accepting requests for jobs within the Work class, if the Work class does not have access to the resources to process them. You also assigned minimum limits to the Listen class to ensure fast response time. The minimum limit for memory ensures that pages used by this class will not be stolen by page replacement, resulting in faster execution time. The minimum limit for CPU ensures that when these processes can be run, they will have the highest-priority access (in the superclass) to the CPU resources.

Step 3. Fine-tune the Workload Manager configuration

1. Monitor the system using the **wlmstat** command and verify that the regulation done by WLM aligns with your goals and does not unduly deprive some applications of resources while others get more than they should. If this is the case, adjust the shares and refresh WLM.
2. As you monitor and adjust the shares, limits, and tier numbers, decide whether you want to delegate the administration for the subclasses for some or all of the superclasses. The administrator can then monitor and set up the subclass shares, limits, and tier number.

The administrator of each superclass can repeat this process for the subclasses of each superclass. The only difference is that WLM cannot run in passive mode at the subclass level only. The subclass configuration and tuning has to be done with WLM in active mode. One way not to impact users and applications in the superclass is to start the tier number, and the shares and limits for the subclasses at their default value ('-' (hyphen) for shares, 0% for minimum, and 100% for soft and hard maximum). With these settings, WLM will not regulate the resource allocation between the subclasses.

For more information

- [Workload Manager](#).
- [Workload Management](#).
- [Workload Management Diagnosis](#) in *Performance management*.
- The descriptions for the [classes](#), [limits](#), [rules](#), and [shares](#) files in the *Files Reference*.
- The **topas**, **wlmassign**, **wlmcheck**, **wlmcntrl**, and **wlmstat**.
- The WLM subroutine descriptions, especially [wlm_set_tag](#).

Classes

Workload Manager helps you control the allocation of system resources by defining classes of service and allocating resources to each of these classes.

Each class has a set of attributes that determine what its resource entitlements are, as well as other behaviors. Every process on the system is classified into a service class, and is thus subject to enforcement of the resource entitlements and behaviors for that class. Processes are assigned to a class either manually using manual assignment, or automatically according to user-defined classification rules.

WLM supports two levels of classes: *superclasses* and *subclasses*. Super classes are given resource entitlements based on available system resources, and subclasses are given resource entitlements relative to the entitlements of their associated superclass. Optionally, you can define subclasses to allow for more granular control of the processes in a superclass. You can also delegate the responsibility of defining subclasses by specifying an admin user or admin group for a superclass.

For both the superclass and subclass levels, you can define classes, resource shares and limits, and rules using SMIT, or the command-line interface. Applications can use the WLM APIs. Configuration definitions are kept in a set of text files called the WLM *property files*.

A class name is up to 16 characters in length and can contain only uppercase and lowercase letters, numbers and underscores (_). For a given WLM configuration, each superclass name must be unique. Each subclass name must be unique within that super classes, but it can match subclass names in other super classes. To uniquely identify every subclass, the full name of a subclass is composed of the superclass name and the subclass name separated by a dot; for example: *Super.Sub*.

Superclasses

The system administrator can define up to 64 superclasses.

In addition, the following five superclasses are automatically created:

Default superclass

Is the default superclass and is always defined. All non-root processes that are not automatically assigned to a specific superclass are assigned to the Default superclass. Other processes can also be assigned to the *Default* superclass by providing specific assignment rules.

System superclass

Has all privileged (root) processes assigned to it if those processes are not assigned by rules to a specific class. This superclass also collects the memory pages belonging to kernel memory segments and kernel processes. Other processes can also be assigned to the System superclass by providing specific assignment rules for this superclass. This superclass has a memory minimum limit of 1% as the default.

Shared superclass

Receives the memory pages that are shared by processes in more than one superclass. This includes pages in shared memory regions and pages in files that are used by processes in more than one superclass (or in subclasses of different superclasses). Shared memory and files that are used by multiple processes that all belong to a single superclass (or subclasses of the same superclass) are associated with that superclass. Only when a process from a different superclass accesses the shared memory region or file are the pages placed in the Shared superclass. This superclass can have only physical memory shares and limits applied to it. It cannot have shares or limits for the other resource

types, subclasses, or assignment rules specified. Whether a memory segment shared by processes in different subclasses of the same superclass is classified into the *Shared* subclass or remains in its original subclass depends on the value of the **localshm** attribute of the original subclass.

Unclassified superclass

Is a memory allocation for unclassified processes. The processes in existence at the time that WLM is started are classified according to the assignment rules of the WLM configuration being loaded. During this initial classification, all the memory pages attached to each process are "charged" either to the superclass that the process belongs to (when not shared, or shared by processes in the same superclass), or to the *Shared* superclass when shared by processes in different superclasses.

However, a few pages cannot be directly tied to any processes (and thus to any class) at the time of this classification, and this memory is charged to the *Unclassified* superclass. Most of this memory is correctly reclassified over time, when it is either accessed by a process, or released and reallocated to a process after WLM is started. There are no processes in the *Unclassified* superclass. This superclass can have physical memory shares and limits applied to it. It cannot have shares or limits for the other resource types, subclasses, or assignment rules specified.

Unmanaged superclass

A special superclass, named *Unmanaged*, is always defined. No processes are assigned to this class. This class accumulates the memory usage for all pinned pages in the system that are not managed by WLM. The CPU utilization for the waitproc processes is not accumulated in any class to prevent the system from appearing to be at 100% utilization. This superclass cannot have shares or limits for any resource types, subclasses, or specified assignment rules.

Subclasses

The system administrator or a superclass administrator can define up to 61 subclasses.

In addition, two special subclasses, *Default* and *Shared*, are always defined.

Default subclass

Is the default subclass and is always defined. All processes that are not automatically assigned to a specific subclass of the superclass are assigned to the *Default* subclass. You can also assign other processes to the *Default* subclass by providing specific assignment rules.

Shared subclass

Receives all the memory pages that are used by processes in more than one subclass of the superclass. Included are pages in shared memory regions and pages in files that are used by processes in more than one subclass of the same superclass. Shared memory and files that are used by multiple processes that all belong to a single subclass are associated with that subclass. Only when a process from a different subclass of the same superclass accesses the shared memory region or file are the pages placed in the *Shared* subclass of the superclass. There are no processes in the *Shared* subclass. This subclass can have only physical memory shares and limits applied to it, and it cannot have shares or limits for the other resource types or assignment rules specified. Whether a memory segment shared by processes in different subclasses of the same superclass is classified into the *Shared* subclass or remains in its original subclass depends on the value of the **localshm** attribute of the original subclass.

Class attributes

List all the attributes of a WLM class.

Class Name

Can be up to 16 characters in length and can only contain uppercase and lowercase letters, numbers and underscores (_).

Tier

A number between 0 and 9 used to prioritize resource allocation between classes.

Inheritance

Specifies whether a child process inherits the class assignment from its parent.

localshm

Prevents memory segments belonging to one class from migrating to Shared class.

Administrator (adminuser, admingroup, authgroup) (superclass only)

Delegates the administration of a superclass.

Authorization (authuser, authgroup)

Delegates the right to manually assign a process to a class.

Resource Set (rset)

Limits the set of resources a given class has access to in terms of CPUs (processor set).

delshm

Deletes the shared memory segments if the last referencing process is killed due to the virtual memory limit.

vmeforce

Indicates whether to kill all processes in a class, or only the offending process, when a class reaches its virtual memory limit.

io_priority

Specifies the priority assigned to I/O requests issued by the threads classified to the class. This priority is used to prioritize I/O buffers at the device level. If the storage device does not support I/O priorities, the priority is ignored. Valid I/O priorities range from 0 to 15.

Related concepts

Process assignment to classes for workload management

The processes are assigned to a class, using class-assignment rules provided by the system administrator. The classification criteria are based on the value of a set of attributes of the process such as user ID, group ID, name of the application file, type of process, and application tag.

Tier attribute

Tiers represent the order in which system resources are allocated to WLM classes.

The administrator can define classes in up to 10 tiers, numbered 0 through 9, with 0 being the highest or most important tier. The amount of resources available to tier 0 is all available system resources. The amount of resources available to the lower (higher number) tiers, is the amount of resources that is unused by all higher tiers. Target consumption percentages for classes are based on the number of active shares in its tier, and the amount of resource available to the tier. Because tier 0 is the only tier that is guaranteed to always have resources available to it, it is recommended that processes that are essential to system operation be classified in a class in this tier. If no tier value is specified for a class, it will be put in tier 0.

A tier can be specified at both the superclass and the subclass levels. Superclass tiers are used to specify resource allocation priority between superclasses. Subclass tiers are used to specify resource allocation priority between subclasses of the same superclass. There is no relationship among sub-tiers of different superclasses.

Inheritance attribute

The inheritance attribute of a class indicates whether processes in the class should be automatically reclassified when one of the classification attributes of the process changes.

When a new process is created with the `fork` subroutine, it automatically inherits its parent's class, whether or not inheritance is enabled. One exception is when the parent process has a tag, has its **inherit tag at fork** set to off, and class inheritance is off for the parent's class. In this case, the child process is reclassified according to the classification rules.

When inheritance is not enabled for a class, any process in the class is automatically classified according to the classification rules after calling any service that changes a process attribute that is used in the rule. The most common of these calls is the `exec` subroutine, but other subroutines that can change classification include `setuid`, `setgid`, `plock`, `setpri`, and `wlm_set_tag`. When inheritance is enabled, the process is not subject to reclassification based on the classification rules, and will remain in its current class. Manual assignment has priority over inheritance and can be used to reclassify processes that are in a class with inheritance enabled.

The specified value for the `inheritance` attribute can be either yes or no. If unspecified, inheritance will not be enabled for a class.

This attribute can be specified at both superclass and subclass level. For a subclass of a given superclass:

- If the `inheritance` attribute is set to yes at both the superclass and the subclass levels, a child of a process in the subclass will remain in the same subclass.
- If the `inheritance` attribute is set to yes for the superclass and no (or unspecified) for the subclass, a child of a process in the subclass will remain in the same superclass and will be classified in one of its subclasses according to the assignment rules for the superclass.
- If the `inheritance` attribute is no (or is unspecified) for the superclass and is set to yes for the subclass, a child of a process in the subclass will be submitted to the automatic assignment rules for the superclasses.
 - If the process is classified by the rules in the same superclass, then it will remain in the subclass (it will not be submitted to the subclass's assignment rules).
 - If the process is classified by the superclass's rules in a different superclass, then the subclass assignment rules of the new superclass are applied to determine the subclass of the new superclass the process will be assigned to.
- If both superclass and subclass `inheritance` attributes are set to no (or are unspecified), then a child of a process in the subclass will be submitted to the standard automatic assignment.

localshm attribute

The `localshm` attribute can be specified at the superclass and the subclass levels.

The `localshm` attribute is used to prevent memory segments belonging to one class from migrating to the *Shared* superclass or subclass when accessed by processes in other classes. The possible values for the attribute are yes or no. A value of yes means that shared memory segments in this class must remain local to the class and not migrate to the appropriate *Shared* class. A value of no is the default when the attribute is not specified.

Memory segments are classified on page faults. When a segment is created, it is marked as belonging to the *Unclassified* superclass. On the first page fault on the segment, this segment is classified into the same class as the faulting process. If, later on, a process belonging to a different class than the segment page faults on this segment, WLM considers whether the segment needs to be reclassified into the appropriate *Shared* class (superclass or subclass). If the faulting process and the segment belong to different superclasses, one of the following occurs:

- If the segment's superclass has the `localshm` attribute set to yes, the segment remains in its current superclass. If the segment's subclass has the `localshm` attribute set to yes, the segment remains in its current subclass. If the superclass `localshm` attribute is set to yes but its subclass attribute is set to no, it goes into the *Shared* subclass of the current superclass.
- If the segment's superclass has the `localshm` attribute set to no, the segment goes to *Shared* superclass. This is the default action.

If the faulting process and the segment belong to different subclasses of the same superclass, and the segment's subclass has the `localshm` attribute set to yes, the segment remains in the current class (superclass and subclass). Otherwise, the segment goes to the *Shared* subclass of the superclass.

Of course, if the faulting process and the segment belong to the same class (same superclass and same subclass), the segment is not reclassified regardless of the values of the `localshm` attributes.

Administrator attribute

The `adminuser` and `admingroup` attributes are used to delegate the superclass administration to a user or group of users.

Note: These attributes are valid only for superclasses.

The `adminuser` attribute specifies the name of the user (as listed in `/etc/passwd`) authorized to perform administration tasks on the superclass. The `admingroup` attribute specifies the name of the group of users (as listed in `/etc/group`) authorized to perform administration tasks on the superclass.

Only one value (user or group) is allowed for each attribute. Either of them, none, or both can be specified. The user or group will have authority to do the following:

- Create and delete subclasses.
- Change the attributes and resource shares and limits for the subclasses.
- Define, remove or modify subclass assignment rules.
- Refresh (update) the active WLM configuration for the superclass.

Authorization attribute

The `authuser` and `authgroup` attributes are valid for all classes. They are used to specify the user or group authorized to manually assign processes to the class (superclass or subclass).

When manually assigning a process (or a group of processes) to a superclass, the assignment rules for the superclass are used to determine to which subclass of the superclass each process will be assigned.

Only one value (user or group) is allowed for each attribute. Either of them, none, or both can be specified.

Resource set attribute

The resource set attribute (called *rset*) can be specified for any class. Its value is the name of a resource set defined by the system administrator.

The *rset* attribute represents a subset of the CPU resource available on the system (processor set). The default is "system," which gives access to all the CPU resources available on the system. The only restriction is that if an *rset* is specified for a subclass, the set of CPUs in the set must be a subset of the CPUs available to the superclass. (For detailed information, see the `mkcrset` command.)

Note: Carefully consider assigning resource sets to any class that is not in tier 0. Because lower tiers only have access to the resources that are unused by the higher tiers, restricting a non-tier-0 class to a subset of the CPUs on the system could result in starvation if there is no CPU time available on those CPUs.

Process classifications in Workload Manager

In WLM, processes can be classified in either of two ways.

- A process is automatically assigned using assignment rules when process classification attributes change. When WLM is running in active mode, this automatic assignment is always in effect (it cannot be turned off). This is the most common way that processes are classified.
- A selected process or group of processes can be manually assigned to a class by a user with the required authority on both the processes and the target class. Manual assignment can be done using by a WLM command, which could be invoked directly or through SMIT or by an application using a function of the WLM Application Programming Interface. This manual assignment overrides the automatic assignment and inheritance.

Automatic class assignment in Workload Manager

The automatic assignment of processes to classes uses a set of class-assignment rules specified by a WLM administrator.

There are two levels of assignment rules:

- A set of assignment rules at the WLM configuration level used to determine which superclass a given process is assigned to.
- Each superclass with subclasses defined, in turn has a set of assignment rules used to determine which subclass of the superclass the process is assigned to.

The assignment rules at both levels are based on the values of a set of process attributes. These attributes are as follows:

- Process user ID
- Process group ID
- Path name of the application (program) executed

- Type of the process (32bit or 64bit, for example)
- Process tag.

The tag is a process attribute, defined as a character string, that an application can set by program, using the WLM API.

The classification is done whenever an attribute changes by comparing the value of these process attributes against lists of possible values given in the class assignment rules file (called `rules`). The comparison determines which rule is a match for the current value of the process attributes.

A class assignment rule is a text string that includes the following fields, separated by one or more spaces:

Item	Description
Name	Must contain the name of a class which is defined in the class file corresponding to the level of the <code>rules</code> file (superclass or subclass). Class names can contain only uppercase and lowercase letters, numbers, and underscores and can be up to 16 characters in length. No assignment rule can be specified for the system defined classes <code>Unclassified</code> , <code>Unmanaged</code> and <code>Shared</code> .
Reserved	Reserved for future extension. Its value must be a hyphen (-), and it must be present.
User	Can contain either a hyphen (-) or at least one valid user name (as defined in the <code>/etc/passwd</code> file). The list is composed of one or more names, separated by a comma (,). An exclamation mark (!) can be used before a name to exclude a given user from the class. Patterns can be specified to match a set of user names, using full Korn shell pattern-matching syntax. If there are no valid user names, the rule is ignored.
Group	Can contain either a hyphen (-) or at least one valid group name (as defined in the <code>/etc/group</code> file). The list is composed of one or more names, separated by a comma (,). An exclamation mark (!) can be used before a name to exclude a given group from the class. Patterns can be specified to match a set of group names using full Korn shell pattern matching syntax. If there are no valid group names, the rule is ignored.
Application	<p>Can contain either a hyphen (-) or a list of application path names. This is the path name of the applications (programs) executed by processes included in the class. The application names will be either full path names or Korn shell patterns that match path names. The list is composed of one or more path names, separated by a comma (,). An exclamation mark (!) can be used before a name to exclude a given application.</p> <p>At least one application in the list must be found at load time or the rule is ignored. Rules that are initially ignored for this reason might become effective later on if a file system is mounted that contains one or more applications in the list.</p>

Item	Description
Type	<p>Can contain either a hyphen (-) or a list of process attributes. The possible values for these attributes are:</p> <ul style="list-style-type: none"> • 32bit: the process is a 32-bit process • 64bit: the process is a 64-bit process • plock: the process called the plock subroutine to pin memory • fixed: the process is a fixed priority process (SCHED_FIFO or SCHED_RR) <p>The fixed type is for classification purposes only. WLM does not regulate the processor use of fixed priority processes or threads. Because fixed priority processes have the potential to cause deprivation among other processes in a class, this classification attribute is provided to allow isolation of these jobs. This attribute can also be used to report consumption of such processes.</p> <p>The value of the type field can be a combination of one or more of the above attributes separated by a plus (+). The 32bit and 64bit values are mutually exclusive.</p>
Tag	<p>May contain either a hyphen (-) or a list of application tags. An application tag is a string of up to 30 alphanumeric characters. The list is composed of one or more application tag values separated by commas.</p>

The User, Group, Application, and Tag attributes can be an attribute value grouping.

When a process is created (fork), it remains in the same class as its parent. Reclassification happens when the new process issues a system call which can modify one of the attributes of the process used for classification; for example, *exec*, *setuid* (and related calls), *setgid* (and related calls), *setpri* and *plock*.

To classify the process, WLM examines the top-level *rules* file for the active configuration to determine which superclass the process belongs. For each rule in the file, WLM checks the current values for the process attributes against the values and lists of values specified in the rule. Rules are checked in the order that they appear in the file. When a match is found, the process is assigned to the superclass named in the first field of the rule. Then the rules file for the superclass is examined in the same way to determine to which subclass the process should be assigned.

For a process to match one of the rules, each of its attributes must match the corresponding field in the rule. The following is a list of the criteria used to determine whether the value of an attribute matches the values in the field of the *rules* file:

- If the field in the rules file has a value of hyphen (-), then any value of the corresponding process attribute is a match.
- For all the attributes except *type*, if the value of the process attribute matches one of the values in the list in the rules file that is not excluded (prefaced by a "!"), then a match has occurred.
- For the *type* attribute, if one of the values in the rule is comprised of two or more values separated by a plus (+), then a process is a match only if its characteristics match all the values.

At both superclass and subclass levels, WLM goes through the rules in the order in which they appear in the *rules* file, and classifies the process in the class corresponding to the first rule for which the process is a match. The order of the rules in the rules file is therefore extremely important. Use caution when you create or modify the rules file.

Manual class assignment in Workload Manager

A process or a group of processes can be manually assigned to a superclass and/or subclass by using SMIT, or the **wlmassign** command.

See **wlmassign** for more information. An application can assign processes through the **wlm_assign** API function.

To manually assign processes to a class or to cancel an existing manual assignment, a user must have the appropriate level of privilege. A manual assignment can be made or canceled separately at the superclass level, the subclass level, or both. This assignment is specified by flags for the programming interface and a set of options for the command line interface used by the WLM administration tools. So, a process can be manually assigned to a superclass only, a subclass only, or to a superclass and a subclass of that superclass. In the latter case, the dual assignment can be done simultaneously (with a single command or API call) or at different times, by different users.

Assignment is very flexible, but can be confusing. Following are two examples of the possible cases.

Example 1: First Assignment of Processes

A system administrator manually assigns *Process1* from *superclassA* to *superclassB* (superclass-level-only assignment). The automatic assignment rules for the subclasses of *superclassB* are used by WLM to determine to which subclass the process is ultimately assigned. *Process1* is assigned to *superclassB.subclassA* and is flagged as having a "superclass only" assignment.

A user with the appropriate privileges assigns *Process2* from its current class *superclassA.subclassA* to a new subclass of the same superclass, *superclassA.subclassB*. *Process2* is assigned to its new subclass and flagged as having a "subclass only" assignment.

A WLM administrator of the subclasses of *superclassB* manually reassigns *Process1* to *subclassC*, which is another subclass of *superclassB*. *Process1* is reclassified into *superclassB.subclassC* and is now flagged as having both superclass and subclass level assignment.

Example 2: Reassignment or Cancellation of Manual Assignment

The reassignment and cancellation of a manual assignment at the subclass level is less complex and affects only the subclass level assignment.

Suppose that the system administrator wants *Process2* to be in a superclass with more resources and decides to manually assign *Process2* to *superclassC*. In Example 1, *Process2* was manually assigned to *subclassB* of *superclassA*, with a "subclass only" assignment. Because *Process2* is assigned to a different superclass, the previous manual assignment becomes meaningless and is canceled. *Process2* now has a "superclass only" manual assignment to *superclassC*, and in the absence of inheritance, is assigned to a subclass of *superclassC* using the automatic assignment rules.

Now, the system administrator decides to terminate the manual assignment from *Process1* to *superclassB*. The "superclass level" manual assignment of *Process1* is canceled, and in the absence of inheritance, *Process1* is assigned a superclass using the top level automatic assignment rules.

If the rules have not changed, *Process1* is assigned to *superclassA*, and its subclass level manual assignment to *superclassB.subclassC* becomes meaningless and is canceled.

If for some reason the top level rules assign *Process1* to *superclassB*, then the subclass level assignment to *superclassB.subclassC* is still valid and remains in effect. *Process1* now has a "subclass only" manual assignment.

Updates to the Workload Manager

When WLM is updated (with the **wlmcntrl -u** command), the updated configuration can load a new set of classification rules.

When this happens, processes are often reclassified using the new rules. WLM does not reclassify processes that have been manually assigned or that are in a class with inheritance enabled, unless their class does not exist in the new configuration.

Security considerations for Workload Manager

To assign a process to a class or to cancel a prior manual assignment, the user must have authority both on the process and on the target class.

These constraints translate into the following rules:

- The root user can assign any process to any class.

- A user with administration privileges on the subclasses of a given superclass (that is, the user or group name matches the user or group names specified in the attributes `adminuser` and `admingroup` of the superclass) can manually reassign any process from one of the subclasses of this superclass to another subclass of the superclass.
- Users can manually assign their own processes (ones associated with the same real or effective user ID) to a subclass for which they have manual assignment privileges (that is, the user or group name matches the user or group names specified in the attributes `authuser` and `authgroup` of the superclass or subclass).

To modify or terminate a manual assignment, users must have at least the same level of privilege as the person who issued the last manual assignment.

Resource management with Workload Manager

WLM monitors and regulates the resource utilization, on a per-class basis, of the threads and processes active on the system. You can set minimum or maximum limits per class for each resource type managed by WLM, as well as a target value per class for each resource.

This target is representative of the amount of the resource that is optimal for the jobs in the class. The shares and limits at the superclass level refer to the total amount of each resource available on the system. At the subclass level, shares and limits refer to the amount of each resource made available to the superclass that the subclass is in (superclass target). The hierarchy of classes is a way to divide the system resources between groups of users (superclasses) and delegate the administration of this share of the resources to superclass administrators. Each superclass administrator can then redistribute this amount of resources between the users in the group by creating subclasses and defining resource entitlements for these subclasses.

Resource types in Workload Manager

WLM manages three types of resources on a percentage consumption basis.

Item	Description
CPU utilization of the threads in a class	This is the sum of all the CPU cycles consumed by every thread in the class.
Physical memory utilization for the processes in a class	This is the sum of all the memory pages which belong to the processes in the class.
Disk I/O bandwidth for the class	This is the bandwidth (in 512-byte blocks per second) of all the I/Os started by threads in the class on each disk device accessed by the class.

Every second, WLM calculates the per-class utilization for each resource during the last second, as a percentage of the total resource available, as follows:

- For the CPU, the total amount of CPU time available every second is equal to 1 second times the number of CPUs on the system. For instance, on an eight-way SMP, if all the threads of a class combined consumed 2 seconds of CPU time during the last second, this represents a percentage of $2/8 = 25\%$. The percentage used by WLM for regulation is a decayed average over a few seconds of this "instantaneous" per-second resource utilization.
- For physical memory, the total amount of physical memory available for processes at any given time is equal to the total number of memory pages physically present on the system minus the number of pinned pages. Pinned pages are not managed by WLM because these pages cannot be stolen from a class and given to another class to regulate memory utilization. The memory utilization of a class is the ratio of the number of non-pinned memory pages owned by all the processes in the class to the number of pages available on the system, expressed as a percentage.
- For disk I/O, the main problem is determining a meaningful available bandwidth for a device. When a disk is 100% busy, its throughput, in blocks per second, is very different if one application is doing sequential I/Os than if several applications are generating random I/Os. If you only used the maximum throughput measured for the sequential I/O case (as a value of the I/O bandwidth available for the

device) to compute the percentage of utilization of the device under random I/Os, you might be misled to think that the device is at 20% utilization, when it is in fact at 100% utilization.

To get more accurate and reliable percentages of per-class disk utilization, WLM uses the statistics provided by the disk drivers (displayed using the **iostat** command) giving for each disk device the percentage of time that the device has been busy during the last second. WLM counts the number of total blocks which have been read or written on a device during the last second by all the classes accessing the device, and how many blocks have been read or written by each class and what was the percentage of utilization of the device. WLM then calculates the percentage of the disk throughput consumed by each class.

For instance, if the total number of blocks read or written during the last second was 1000 and the device had been 70% busy, this means that a class reading or writing 100 blocks used 7% of the disk bandwidth. Similarly to the CPU time (another renewable resource), the values used by WLM for its disk I/O regulation are also a decayed average over a few seconds of these per second percentages.

For the disk I/O resource, the shares and limits apply to each disk device individually accessed by the class. The regulation is done independently for each device. This means that a class could be over its entitlement on one device, and the I/Os to this device will be regulated while it is under its entitlement on another disk and the I/Os to this other device will not be constrained.

WLM supports accounting and regulation of resources on the basis of total consumption. There are two types of resources that can be regulated in this manner: class totals and process totals.

class totals

Per-class limits can be specified for the number of processes, threads, and login sessions in the class. These limits are specified as the absolute number of each resource that can exist in the class at any instant. These limits are strictly enforced; when a class has reached its limit for one of these resources, any attempt to create another instance of the resource will fail. The operation will continue to fail for any process in the class until the class is below its specified limit for the resource.

process totals

Per-process limits can be specified for the total amount of CPU time, blocks of disk I/O, and connect time for a login session. These limits are specified at the class level, but apply to each process in the class individually (each process can consume this amount). These consumption values are cumulative and thus represent the total amount of each particular resource that has been consumed by the process during its lifetime. Once a process has exceeded its total limit for any resource, the process will be terminated. The process will be sent a SIGTERM signal, and if it catches this signal and does not exit after a 5 second grace period, will be sent a SIGKILL signal. When a login session has reached 90% of its connect time limit, a warning message will be written to the controlling terminal to warn the user that the session will soon be terminated.

Target shares in Workload Manager

The target (or desired) resource consumption percentage for a class is determined by the number of shares it has for a particular resource.

The shares represent how much of a particular resource a class should get, relative to the other classes in its tier. A class's target percentage for a particular resource is simply its number of shares divided by the number of active shares in its tier. If limits are also being used, the target is limited to the range [minimum, soft maximum]. If the calculated target outside this range, it is set to the appropriate upper/lower bound (see Resource Limits). The number of active shares is the total number of shares of all classes that have at least one active process in them. Because the number of active shares is dynamic, so is the target. If a class is the only active class in a tier, its target will be 100% of the amount of resource available to the tier.

For example, assume three active superclasses classes in tier 0—A, B, and C—with shares for a particular resource of 15, 10, and 5, respectively, the targets would be:

target(A) = 15/30 = 50%
target(B) = 10/30 = 33%
target(C) = 5/30 = 17%

If some time later, class B becomes inactive (no active processes), the targets for class A and C will be automatically adjusted:

$$\text{target(A)} = 15/20 = 75\%$$
$$\text{target(C)} = 5/20 = 25\%$$

As you can see, the shares represent a self-adapting percentage which allow the resources allocated to a class to be evenly distributed to or taken from the other classes when it becomes active/inactive.

To allow a high degree of flexibility, the number of shares for a class can be any number between 1 and 65535. Shares can be specified for superclasses and subclasses. For superclasses, the shares are relative to all other active superclasses in the same tier. For subclasses, the shares are relative to all other active subclasses in the same superclass, in the same tier. The shares for a subclass one superclass have no relationship to the shares for a subclass of another superclass.

In some cases, it might be desirable to make the target for a class independent from the number of active shares. To accomplish this, a value of "-" can be specified for the number of shares. In this case, the class will be unregulated for that resource, meaning that it has no shares, and its target is not dependent on the number of active shares. Its target will be set to (resource available to the tier - the sum of mins for all other classes in the tier). This target, or actual consumption (whichever is lower) is then taken out of what is available to other classes in the same tier.

For example, assume classes A, B, C, and D have shares for a particular resource of "-", 200, 150, and 100, respectively. All classes are active, and class A is consuming 50% of the resource:

$$\text{target(A)} = \text{unregulated} = 100\%$$
$$\text{target(B)} = 200/450 * \text{available} = 44\% * 50\% = 22\%$$
$$\text{target(C)} = 150/450 * \text{available} = 33\% * 50\% = 17\%$$
$$\text{target(D)} = 100/450 * \text{available} = 22\% * 50\% = 11\%$$

Because class A is unregulated and is consuming 50% of the available resource, the other classes only have 50% available to them, and their targets are calculated based on this percentage. Since class A will always be below its target (100%), it will always have a higher priority than all other classes in the same tier that are at or above their targets (see [“Class priority in Workload Manager”](#) on page 157 for more information).

Note: Making a class unregulated for a resource is not the same as putting it in a higher tier. The following behaviors, listed here, are true for an unregulated class (in the same tier), and are not true if the class is put in a higher tier:

- Because the shares are defined on a per-resource basis, a class can be unregulated for one or more resources, and regulated for others.
- The minimum limits for other classes in the same tier are honored. Higher tiers do not honor minimums specified in the lower tiers.
- Even in the absence of minimum limits for the classes with shares, the consumption of unregulated classes is somewhat dependent on classes with shares since they are competing for some of the resource available to the tier. Some experimentation is required to see what the behavior is with a given workload.

If the number of shares is unspecified, a default value of "-" will be used, and the class will be unregulated for that resource. Note that in the first version of WLM, the default share value, if unspecified, was 1.

The shares are specified per class for all the resource types. Shares are specified in stanzas of the **shares** file. For example:

```
shares
classname:
  CPU      = 2
  memory   = 4
  diskIO   = 3
```

Resource limit specification in Workload Manager

In addition to using shares to define relative resource entitlements, WLM provides the ability to specify resource limits for a class. Resource limits allow the administrator to have more control over resource allocation. These limits are specified as percentages and are relative to the amount of resource available to the tier that the class is in.

There are three type of limits for percentage-based regulation:

Minimum

This is the minimum amount of a resource that should be made available to the class. If the actual class consumption is below this value, the class will be given highest priority access to the resource. The possible values are 0 to 100, with 0 being the default (if unspecified).

Soft Maximum

This is the maximum amount of a resource that a class can consume when there is contention for that resource. If the class consumption exceeds this value, the class will be given the lowest priority in its tier. If there is no contention for the resource (from other classes in the same tier), the class will be allowed to consume as much as it wants. The possible values are 1 to 100, with 100 being the default (if unspecified).

Hard Maximum

This is the maximum amount of a resource that a class can consume, even when there is no contention. If the class reaches this limit, it will not be allowed to consume any more of the resource until its consumption percentage falls below the limit. The possible values are 1 to 100, with 100 being the default (if unspecified).

Resource limit values are specified in the resource limit file by resource type within stanzas for each class. The limits are specified as a minimum to soft maximum range, separated by a hyphen (-) with white space ignored. The hard maximum when specified follows the soft maximum, and is separated by a semicolon (;). Each limit value is followed by a percent sign (%).

The following are examples using the rules files:

- If user joe from group acct3 executes `/bin/vi`, then the process will be put in superclass acctg.
- If user sue from group dev executes `/bin/emacs`, then the process will be in the superclass devlt (group ID match), but will not be classified into the editors subclass, because user sue is excluded from that class. The process will go to devlt. Default.
- When a DB administrator starts `/usr/sbin/oracle` with a user ID of `oracle` and a group ID of `dbm`, to serve the database DB1, the process is classified in the default superclass. Only when the process sets its tag to `_DB1` will it be assigned to superclass db1.

Limits are specified for all resource types, per class, in stanzas of the `limits` file. For example:

```
shares
classname:
  CPU      =    0%-50%;80%
  memory   =   10%-30%;50%
```

In this example, no limits are set for disk I/O. Using the system defaults, this translates to the following:

```
diskIO    =    0%-100%;100%
```

All of the preceding examples assume that the superclasses and subclasses described do not have the inheritance attribute set to yes. Otherwise, the new processes would simply inherit the superclass or subclass from their parent.

The following are the only constraints that WLM places on resource limit values:

- The minimum limit must be less than or equal to the soft maximum limit.
- The soft maximum limit must be less than or equal to the hard maximum limit.
- The sum of the minimum of all the superclasses within a tier cannot exceed 100.
- The sum of the minimum of all the subclasses of a given superclass within a tier cannot exceed 100.

When a class with a hard memory limit has reached this limit and requests more pages, the VMM page replacement algorithm (LRU) is initiated and "steals" pages from the limited class, thereby lowering its number of pages below the hard maximum, before handing out new pages. This behavior is correct, but extra paging activity, which can take place even where there are plenty of free pages available, impacts the general performance of the system. Minimum memory limits for other classes are recommended before imposing a hard memory maximum for any class.

Because classes under their minimum have the highest priority in their tier, the sum of the minimums should be kept to a reasonable level, based on the resource requirements of the other classes in the same tier.

The constraint of the sum of the minimum limits within a tier being less than or equal to 100 means that a class in the highest priority tier is always allowed to get resources up to its minimum limit. WLM does not guarantee that the class will actually reach its minimum limit. This depends on how the processes in the class use their resources and on other limits that are in effect. For example, a class might not reach its minimum CPU entitlement because it cannot get enough memory.

For physical memory, setting a minimum memory limit provides some protection for the memory pages of the class's processes (at least for those in the highest priority tier). A class should not have pages stolen when it is below its minimum limit unless all the active classes are below their minimum limit and one of them requests more pages. A class in the highest tier should never have pages stolen when it is below its minimum. Setting a memory minimum limits for a class of interactive jobs helps make sure that their pages will not all have been stolen between consecutive activations (even when memory is tight) and improves response time.



Attention: Using hard maximum limits can have a significant impact on system or application performance if not used appropriately. Because imposing hard limits can result in unused system resources, in most cases, soft maximum limits are more appropriate. One use for hard maximum limits might be to limit the consumption of a higher tier to make some resource available to a lower tier, though putting applications that need resources in the higher tier may be a better solution.

The total limits can be specified in the limits file with values and units summarized in the following table:

<i>Table 10. Resource Limits for Workload Manager</i>				
Resource	Allowed Units	Default Unit	Maximum Value	Minimum Value
totalCPU	s, m, h, d, w	s	$2^{30} - 1s$	10 s
totalDiskIO	KB, MB, TB, PB, EB	KB	$(2^{63} - 1) * 512/1024$ KB	1 MB
totalConnectTime	s, m, h, d, w	s	$2^{63} - 1$ s	5 m
totalProcesses	–	–	$2^{63} - 1$	2
totalThreads	–	–	$2^{63} - 1$	2
totalLogins	–	–	$2^{63} - 1$	1

Note: The unit specifiers are not case sensitive. s = seconds, m = minutes, h = hours, d = days, w = weeks, KB = kilobytes, MK = megabytes, ... etc.

An example limits stanza follows:

```
BadUserClass:
    totalCPU = 1m
    totalConnectTime = 1h
```

The total limits can be specified using any value in the above table with the following restrictions:

- If specified, the value for totalThreads must be at least the value of totalProcesses.
- If totalThreads is specified and totalProcesses is not, the limit for totalProcesses will be set to the value of totalThreads.

The total limits can be specified at the superclass and subclass level. When checking the limits, the subclass limit is checked before the superclass limit. If both limits are specified, the lower of the two is enforced. If the specified subclass limit is greater than its associated superclass limit, a warning will be issued when the configuration is loaded, but it will be loaded. This is significant for the class total limits since the limit is absolute (not relative to the superclass) and one subclass could consume all resources available to the superclass. If unspecified, the default value for all total limits is "-" which means no limit. By default, class and process total accounting and regulation will be enabled when WLM is running. The **-T [class|proc]** option of the **wlmcntrl** command can be used to disable total accounting and regulation.

Class priority in Workload Manager

WLM allocates resources to classes by giving a priority to each class for each resource.

The priority of a class is dynamic and is based on the tier, shares, limits, and the current consumption of the class. At any given time, the highest priority class or classes will be given preferential access to resources. At the highest level, tiers represent non-overlapping ranges of class priority. Classes in tier 0 will always be guaranteed to have a higher priority than classes in tier 1 (unless over hard maximum).

When determining class priority, WLM enforces its constraints with the following precedence (highest to lowest):

hard limits

If class consumption exceeds the hard maximum for a resource, the class is given the lowest-possible priority for the resource and will be denied access until its consumption falls below this limit.

tier

In the absence of hard limits, a class's priority will be bounded by the minimum and maximum allowed priority for its tier.

soft limits

If class consumption is below the minimum of the soft maximum limits for a resource, the class is given the highest-priority in the tier. If class consumption is above the soft maximum, the class is given the lowest-priority in the tier.

shares

These are used to calculate the class consumption targets for each resource. The class priority is increased as class consumption falls below the target, and is decreased as it rises above the target. Note that soft limits have higher precedence and the class's priority will be determined based on them, when applicable.

Even though both shares and limits can be used for each class and for each resource, results are more predictable if only one or the other is used per class.

Exclusive use processor resource sets

Exclusive use processor resource sets (XRSETs) allow administrators to guarantee resources for important work. An XRSET is a named resource set that changes the behavior of all the CPUs that it includes. Once a CPU is exclusive, it only runs programs explicitly directed to it.

Creating an XRSET

You must be a root user to create an XRSET. Use the **mkrset** command to create a resource set in the **sysxrset** namespace. For example, the command **mkrset -c 1-3 sysxrset/set1** creates an XRSET for CPUs 1, 2, and 3. The **rs_registername()** subroutine can also be used to create an XRSET.

Determining if XRSETs have been Defined on a System

The **lsrset -v -n sysxrset** command displays all XRSETs defined on a system. (There is currently no programming API to do this.)

Deleting an XRSET

You must be a root user to delete an XRSET. The **rmrset** command deletes an XRSET. The **rs_discardname()** subroutine may also be used to delete an XRSET.

Rebooting the System

When you reboot the system, any XRSETs that were set are removed from the registry and are no longer in effect.

Specifying Work for XRSETs

There are multiple ways for work to be marked as eligible to use exclusive use processors. The **attachrset** and **execrset** commands can be used to specify resource sets that contain exclusive use processors. Resource sets containing exclusive use processors can be associated with WLM classes. Work classified into such WLM classes will use the exclusive use processors specified in the resource set.

Using XRSETs with Bindprocessor and _system_configuration.ncpus

You cannot use `bindprocessor` to cause work to run on exclusive use processors. Only resource set-based attachments can cause work to run on exclusive use processors.

The number of CPUs in the system configuration (the **_system_configuration.ncpus** field) is not changed when XRSETs are created. There are still NCPUs in the system.

When programs use the `bindprocessor` system call to NCPUs, CPUs in XRSETs will fail with the EINVAL error. You can bind to any ID returned by the query option of the **bindprocessor** command. The query option (`bindprocessor -q`) will only return valid bind IDs, excluding those that are associated with exclusive CPUs.

For example, if there are 10 CPUs online in a system and three of them are in XRSETs, a `bindprocessor` to CPUs with bind IDs in the range of 0 to 6 will succeed. A `bindprocessor` to CPUs with bind IDs in the range of 7 to 9 will receive an EINVAL error.

Using XRSETs with Dynamic CPU Reconfiguration Operations

In general, dynamic CPU reconfiguration is not affected by exclusive use processors. However, the creation of XRSETs and assignment of work to those processors may prevent removal of a CPU. CPUs that are dynamically added to the system may enter the system as either general use or exclusive use processors. They will enter the system as exclusive use if there is an XRSET containing the logical CPU ID when it enters the system.

Resource sets in Workload Manager

WLM uses resource sets (or *rsets*) to restrict the processes in a given class to a subset of the system's physical resources. In WLM, the physical resources managed are the memory and the processors. A valid resource set is composed of memory and at least one processor.

Using SMIT a system administrator can define and name resource sets containing a subset of the resources available on the system. Then, using the WLM administration interfaces, root user or a designated superclass administrator can use the name of the resource set as the `rset` attribute of a WLM class. From then on, every process assigned to this WLM class is dispatched only on one of the processors in the resource set, effectively separating workloads for the CPU resource.

All of the current systems have only one memory domain shared by all the resource sets, so this method does not physically separate workloads in memory.

Resource set registry in Workload Manager

The `rset` registry services enable system administrators to define and name resource sets so that they can then be used by other users or applications.

To alleviate the risks of name collisions, the registry supports a two-level naming scheme. The name of a resource set is in the form *name_space/rset_name*. Both the *name_space* and *rset_name* can each be 255 characters in length, are case-sensitive, and may contain only uppercase and lowercase letters, numbers, underscores, and periods (.). The *name_space* of **sys** is reserved by the operating system and used for `rset` definitions that represent the resources of the system.

The `rset` definition names are unique within the registry name space. Adding a new `rset` definition to the registry using the same name as an existing `rset` definition causes the existing definition to be

replaced with the new definition, given the proper permission and privilege. Only root can create, modify, and delete resource sets and update the in-core rset database, using SMIT.

Each rset definition has an owner (user ID), group (group ID), and access permissions associated with it. These are specified at the time the rset definition is created and exist for the purpose of access control. As is the case for files, separate access permissions exist for the owner, group and others that define whether read and/or write permission has been granted. Read permission allows an rset definition to be retrieved while write permission allows an rset definition to be modified or removed.

The rset definitions defined by the system administrators are kept in the `/etc/rsets` stanza file. The format of this file is not described, and users must manipulate rset through the SMIT interfaces to prevent future potential compatibility problems if the file format is modified. As is the case for WLM class definitions, the rset definitions must be loaded into kernel data structures before they can be used by WLM.

Setting up Workload Manager

Class definitions, class attributes, the shares and limits, and the automatic class assignment rules, can be entered using SMIT, or the WLM command-line interface. These definitions and rules are kept in plain text files, that can also be created or modified using a text editor.

These files (called *WLM property files*) are kept in the subdirectories of `/etc/wlm`. A set of files describing super classes and their associated subclasses define a WLM configuration. The files for the WLM **Config** configuration are in `/etc/wlm/Config`. This directory contains the definitions of the WLM parameters for the super classes. The files are named `description`, `classes`, `shares`, `limits`, and `rules`. This directory might also contain subdirectories with the name of the super classes where the subclass definitions are kept. For example, for the superclass *Super* of the WLM configuration **Config**, the directory `/etc/wlm/Config/Super` contains the property files for the subclasses of the superclass *Super*. The files are named `description`, `classes`, `shares`, `limits`, and `rules`.

After a WLM configuration has been defined by the system administrator, it can be made the active configuration using the **smnit wlmmanage** fast path, or the **wlmcntrl** command.

You can define multiple sets of property files, defining different configurations of workload management. These configurations are usually located in subdirectories of `/etc/wlm`. A symbolic link `/etc/wlm/current` points to the directory containing the current configuration files. This link is updated by the **wlmcntrl** command when WLM starts with a specified set of configuration files.

Application requirements for Workload Manager configuration

The first phase of defining a configuration requires an understanding of your users and their computing needs, as well as the applications on your system, their resource needs and the requirements of your business (for example, which tasks are critical and which can be given a lower priority). Based on this understanding, you define your super classes and then your subclasses.

Setting priorities is dependent on what function WLM serves in your organization. In the case of server consolidation, you might already know the applications, the users and their resource requirements, and you might be able to skip or shorten some of the steps.

WLM allows you to classify processes by user or group, application, type, tag, or a combination of these attributes. Because WLM regulates the resource utilization among classes, system administrators should group applications and users with the same resource utilization patterns into the same classes. For instance, you might want to separate the interactive jobs that typically consume very little CPU time but require quick response time from batch-type jobs that typically are very CPU- and memory-intensive. This is the same in a database environment in which you need to separate the OLTP-type traffic from the heavy-duty queries of data mining.

This step is done using SMIT, or command-line interface. For the first few times, it is probably a good idea to use SMIT to take you through the steps of creating your first WLM configuration, including defining the super classes and setting their attributes. For the first pass, you can set up some of the attributes and leave the others at their default value. This is the same for the resource shares and limits. All these class characteristics can be dynamically modified at a later time.

You can then start WLM in passive mode, check your classification, and start reviewing the resource utilization patterns of your applications.

Verify your configuration, using the **wlmcheck** command or the corresponding SMIT menus. Then start WLM in passive mode on the newly defined configuration. WLM will classify all the existing processes (and all processes created from that point on) and start compiling statistics on the CPU, memory, and disk I/O utilization of the various classes. WLM will not try to regulate this resource usage.

Verify that the various processes are classified in the appropriate class as expected by the system administrator (using the **-o** flag of the **ps** command). If some of the processes are not classified as you expect, you can modify your assignment rules or set the inheritance bit for some of the classes (if you want the new processes to remain in the same class as their parent) and update WLM. You can repeat the process until you are satisfied with this first level of classification (super classes).

Running WLM in passive mode and refreshing WLM (always in passive mode) involves low risk, has low overhead operation, and can be done safely on a production system without disturbing normal system operation. To activate and refresh WLM, use the **wlmcntrl** command, invoked either from the command line or from SMIT.

Run WLM in passive mode to gather statistics by using the **wlmstat** command. The **wlmstat** command can be used at regular time intervals to display the per-class resource utilization as a percentage of the total resource available, for super classes). This allows you to monitor your system for extended periods of time to review the resource utilization of your main applications.

Tiers, shares, and limits in Workload Manager

Using the data gathered from running WLM in passive mode and your business goals, decide which tier number will be given to every superclass and what share of each resource should be given to the various classes.

For some classes, you might want to define minimum or maximum limits. Adjust the shares and tier numbers to reach your resource-allocation goals. Reserve limits for cases that cannot be solved only with shares. Also, you might decide you need to add subclasses.

- Use minimum limits for applications that typically have low resource usage but need a quick response time when activated by an external event. One of the problems faced by interactive jobs in situations where memory becomes tight is that their pages get stolen during the periods of inactivity. A memory minimum limit can be used to protect some of the pages of interactive jobs if the class is in tier 0.
- Use maximum limits to contain some resource-intensive, low-priority jobs. Unless you partition your system resources for other reasons, a hard maximum will make sense mostly for a nonrenewable resource such as memory. This is because of the time it takes to write data out to the paging space if a higher priority class needs pages that the first class would have used. For CPU usage, you can use tiers or soft maximum to make sure that if a higher priority class is immediately assigned CPU time.

When creating and adjusting the parameters of subclasses, you can refresh WLM only for the subclasses of a given superclass that do not affect users and applications in the other super classes, until you are satisfied with the system behavior.

You can also define other configurations with different parameters, according to the needs of the business. When doing so, you can save time by copying and modifying existing configurations.

Fine-tuning the Workload Manager configuration

Monitor the system using the **wlmstat** command and verify that the regulation done by WLM aligns with your goals and does not unduly deprive some applications of resources while others get more than they should. If this is the case, adjust the shares and refresh WLM.

As you monitor and adjust the shares, limits, and tier numbers, decide whether you want to delegate the administration for the subclasses for some or all of the super classes. The administrator can then monitor and set up the subclass shares, limits, and tier number.

The administrator of each superclass can repeat this process for the subclasses of each superclass. The only difference is that WLM cannot run in passive mode at the subclass level only. The subclass

configuration and tuning has to be done with WLM in active mode. One way not to impact users and applications in the superclass is to start the tier number, and the shares and limits for the subclasses at their default value ('-' (hyphen) for shares, 0% for minimum, and 100% for soft and hard maximum). With these settings, WLM will not regulate the resource allocation between the subclasses.

Troubleshooting Workload Manager

If you are not seeing the desired behavior with your current configuration, you might need to adjust your WLM configuration.

The consumption values for each class can be monitored using the **wlmstat** command. This data can be collected and analyzed to help determine what changes might need to be made to the configuration. After you update the configuration, update the active WLM configuration using the **wlmcntrl -u** command.

The following guidelines can help you decide how to change your configuration:

- If the number of active shares in a tier varies greatly over time, you can give a class no shares for a resource so it can have a consumption target that is independent from the number of active shares. This technique is useful for important classes that require high-priority access to a resource.
- If you need to guarantee access to a certain amount of a resource, specify minimum limits. This technique is useful for interactive jobs that do not consume a lot of resources, but must respond quickly to external events.
- If you need to limit access to resources but shares do not provide enough control, specify maximum limits. In most cases, soft maximum limits are adequate, but hard maximums can be used for strict enforcement. Because hard maximum limits can result in wasted system resources, and they can increase paging activity when used for memory regulation, you should impose minimum limits for the other classes before imposing any hard limits.
- If less-important jobs are interfering with more-important jobs, put the less-important jobs in a lower tier. This technique ensures less-important jobs have lower priority and cannot compete for available resources while the more-important jobs are running.
- If a class cannot reach its consumption target for a resource, check whether this condition is caused by contention for another resource. If so, change the class allocation for the resource under contention.
- If processes within a class vary greatly in their behaviors or resource consumption, create more classes to gain more granular control. Also, it might be desirable to create a separate class for each important application.
- If your analysis shows the resource required by one class is dependent on the consumption of another class, reallocate your resources accordingly. For example, if the amount of resource required by ClassZ is dependent on the number of work requests that can be handled by ClassA, then ClassA must be guaranteed access to enough resources to provide what ClassZ needs.
- If one or more applications are consistently not receiving enough resources to perform adequately, your only option might be to reduce the workload on the system.

Note: You can define an *adminuser* for a superclass to reduce the amount of work that is required of the WLM administrator. After the top-level configuration has been tested and tuned, subsequent changes (including creating and configuring subclasses) can be made by the superclass adminusers to suit their particular needs.

Workload Manager API

Applications can use the WLM APIs, a set of routines in the `/usr/lib/libwlm.a` library, to perform all the tasks that a WLM administrator can perform using the WLM command-line interface.

These include:

- Create, modify, or delete classes
- Change class attributes or resource shares and limits
- Removing classes
- Manually assign processes to classes

- Retrieving WLM statistics

The API allows applications to set an application-defined classification attribute called a tag. Setting this tag using a set of values provided by the system administrator (through the application user documentation) allows discrimination between several instances of the same application. The different classes can therefore be classified with different resource entitlements.

In addition, the **wlm_set_tag** routine allows an application to set up an application tag and specify whether this tag should be inherited by child processes at **fork** or **exec**. Threads can also be assigned application tags with the **wlm_set_thread** tag. A thread's application tag can be inherited across the **fork**, **exec** or **pthread_create** subroutines. The library provides support for multi-threaded 32-bit or 64-bit applications.

Application tag

The application tag is a string of characters and is used as one of the classification criteria for the automatic classification of processes or threads (using the *rules* file). This tag basically provides an application-defined classification criteria in addition to the system-defined criteria such as *user*, *group*, *application* and *type*.

When an application process or thread sets its tag, it is immediately reclassified using the superclass and subclass rules in effect for the currently active WLM configuration. WLM then reviews the assignment rules looking for a match, using all the process attributes, including the new tag.

To be effective, this tag must appear in one or more of the assignment rules. The format and the use of the various tags by each application must be clearly specified in the application's administration documentation and well-known to the WLM administrators so that they can use the various values of the tags in their assignment rules to distinguish between different instances of the same application.

Because different users might have different requirements regarding what set of characteristics of the application processes they want to use to classify those processes, it is recommended that the application provide a set of configuration or run time attributes that can be used to build the tag. The application administrator can specify the format of this tag to the application. The attributes that can be used for the tag and the syntax to specify the format of the WLM tag are application-dependent and are the responsibility of the application provider.

For example, an instance of a database server is able to determine which database it is working on (*db_name*) and through which TCP port a given user is connected (*port_num*). Administrators might have any of the following priorities:

- To create different classes for processes accessing different databases to give each class different resource entitlement
- To separate the processes serving remote requests from different origins and to use the port number as a classification attribute
- To create one superclass for each database and subclass per port number in each superclass.

One way to accommodate these different needs is to specify the content and format of the tag. In this example, assume that the tag can be passed to the application in a configuration file or run time parameter such as `WLM_TAG=$db_name` or `WLM_TAG=$db_name_$port_num`.

When setting its tag, an application can specify whether this tag is inherited by its children so that all the processes created by a specific instance of an application can be classified in the same class. Setting the tag inheritance is how the application tag is most commonly used.

The following is an example of how application tags could be used. In this example, the tag of the database is the same as the database name. Then two instances of the server working on two different databases would set up two different tags, for instance `db1` and `db2`.

A system administrator could create two different classes `dbserv1` and `dbserv2` and classify the two database servers (and all their children if tag inheritance is used) in these classes using the tags. It would then be possible to give each class different resource entitlement according to specific business goals.

The corresponding assignment rules could look similar to the following:

* class	resvd	user	group	application	type	tag
* dbserv1	-	-	dbadm	/usr/sbin/dbserv	-	db1
dbserv2	-	-	dbadm	/usr/sbin/dbserv	-	db2

Application programming interface types

The following are the Workload Manager (WLM) application programming interface types.

Class Management APIs

The WLM API provides applications with the ability to:

- Query the names and characteristics of the existing classes of a given WLM configuration (`wlm_read_classes`).
- Create a new class for a given WLM configuration, define the values of the various attributes of the class (such as tier and inheritance) and the shares and limits for the resources managed by WLM, such as CPU, physical memory, and block I/O (`wlm_create_class`).
- Change the characteristics of an existing class of a given WLM configuration, including the class attributes and resource shares and limits (`wlm_change_class`).
- Delete an existing class of a given configuration (`wlm_delete_class`).

The changes will be applied only to the property files of the specified WLM configuration. Optionally, by specifying an empty string as the configuration name, it is possible to apply the change only to the in-core classes, resulting in an immediate update of the active configuration state.

The API calls require from the caller the same level of privilege that would be required for the command line or for the SMIT interface, as follows:

- Any user can read the class names and characteristics
- Only the root user can create, modify, or delete superclasses
- Only the root user or designated superclass administrators (superclass attributes `adminuser` or `admingroup`) can create, modify, or delete subclasses of a given superclass.

In cases where WLM administration is accomplished, both through the command line and administration tools by WLM administrators, and by application(s) through the API, some caution must be applied. Both interfaces share the same name space for the superclass and subclass names, and the total number of superclasses and subclasses.

In addition, when the API directly modifies the in-core WLM data (create new classes, for example), WLM administrators are not aware of this until classes that they did not create appear on the output of commands such as the **wlmstat** command. To avoid conflicts that would confuse the applications using this API when the system administrator updates WLM, the classes created through the API that are not defined in the WLM property files are not automatically removed from the in-core data. They remain in effect until explicitly removed through the `wlm_delete_class` routine or through an invocation of the **rmclass** command (invoked directly or through SMIT by the system administrator).

The WLM API also provides applications with the ability to:

- Query or change the mode of operation of WLM using the `wlm_set` function
- Query the current status of WLM
- Stop WLM
- Toggle between active to passive mode
- Turn the **rset** binding on and off
- Start or update WLM with the current or an alternate configuration by using the `wlm_load` routine
- Assign a process or a group of processes to a class by using the `wlm_assign` routine.

The API requires the same levels of privilege as the corresponding **wlmcntrl** and **wlmassign** commands:

- Any user can query the state of WLM
- Only the root user can change the mode of operation of WLM
- Only the root user can update or refresh a whole configuration
- root or an authorized superclass administrator (adminuser or admingroup) can update WLM for the subclasses of a given superclass
- root, an authorized user (specified by authuser or authgroup), or an authorized superclass administrator (adminuser or admingroup) can assign processes to a superclass or subclass.

WLM Statistics API

The WLM API routines and `wlm_get_bio_stats` provide application access to the WLM statistics displayed by the **wlmstat** commands.

WLM Classification API

The `wlm_check` routine allows the user to verify the class definitions and the assignment rules for a given WLM configuration. The API routine `wlm_classify` allows an application to determine to which class a process with a specified set of attributes would be classified.

Binary compatibility

To provide binary compatibility if there are future changes to the data structures, each API call is passed a version number as a parameter.

This allows the library to determine with which version of the data structures the application has been built.

Examples of Workload Manager classification, rules, and limits

Several methods exist for classifying a process, and all operate concurrently.

A top-down strictest first-matching algorithm is used to provide for maximum configuration flexibility. You can organize process groupings by user with special cases for programs with certain names, by pathname with special cases for certain users, or any other arrangement.

Example of Workload Manager assignment rules

This example shows a top-level rules file for the configuration *Config* (/etc/wlm/Config/rules file).

```
* This file contains the rules used by WLM to
* assign a process to a superclass
*
* class resvd user      group  application  type  tag
db1          -          -      /usr/bin/oracle*  -    _DB1
db2          -          -      /usr/bin/oracle*  -    _DB2
devlt        -          -      dev            -    -
VPs          -    bob,tcd  -            -    -
acctg        -          -      acct*         -    -
System       -    root    -            -    -
Default      -          -      -            -    -
```

The following is an example of the rules file for the devlt superclass in the /etc/wlm/Config/devlt/rules file:

```
* This file contains the rules used by WLM to
* assign a process to a subclass of the
* superclass devlt
*
* class resvd user      group  application  type  tag
hackers      -    jim,liz  -            -    -
hogs         -          -      -            64bit+plck  -
editors      -    !sue    -      /bin/vi,/bin/emacs  -    -
build        -          -      /bin/make,/bin/cc   -    -
Default      -          -      -            -    -
```

Note: The asterisk (*) is the comment character used in the rules file.

The following are examples using this rules file. The following examples assume that the superclasses and subclasses described do not have the inheritance attribute set to yes. If inheritance were enabled, new processes would inherit the superclass or subclass from their parent processes.

- If user joe from group acct3 executes **/bin/vi**, then the process will be put in superclass acctg.
- If user sue from group dev executes **/bin/emacs**, then the process will be put in the superclass devlt (group ID match), but will not be classified into the editors subclass, because the user is excluded from that class. The process will go to devlt by default.
- When a database administrator with a user ID of oracle and a group ID of dbm starts **/usr/sbin/oracle** to serve the DB1 database, the process will be classified in the Default superclass. Only when the process sets its tag to _DB1 will it be assigned to the superclass db1.

Example of Workload Manager classes with shares and limits

For this example, assume classes A, B, C, and D have shares of 3, 2, 1, and 1 respectively.

If classes A, C, and D are active, the calculated targets would be:

target(A) = $3/5 = 60\%$
target(C) = $1/5 = 20\%$
target(D) = $1/5 = 20\%$

If during testing it was found that the applications in class A perform adequately when allowed to use 50% of the resource, it might be desirable to make the other 50% of the resource available to the other classes. This can be accomplished by giving class A a soft maximum of 50% for this resource. Since the current calculated target of 60% is over this limit, it will be adjusted down to the soft maximum value. When this happens, the target or actual consumption (whichever is lower) of class A is subtracted from the amount of resource available. Because this class now has a target constrained by its limit (and not its shares), the shares for the class are also subtracted from the number of active shares. Assuming class A has a current consumption of 48%, the targets will now be:

target(A) = $3/5 = 60\%$, softmax = 50, = 50%
target(C) = $1/2 * (100 - 48) = 26\%$
target(D) = $1/2 * (100 - 48) = 26\%$

Some time later, all classes may become active, and the targets will again be automatically adjusted:

target(A) = $3/7 = 42\%$
target(B) = $2/7 = 28\%$
target(C) = $1/7 = 14\%$
target(D) = $1/7 = 14\%$

Example of Workload Manager classes with CPU limits

This example examines CPU allocation, assuming that each class can consume all the CPU that it is given.

Two classes, A and B, are in the same tier. CPU limits for A are [30% - 100%]. CPU limits for B are [20% - 100%]. When both classes are running and are using sufficient CPU, WLM first makes sure that they both get their minimum percentages of each second (averaged over several seconds). Then WLM distributes the remaining CPU cycles according to any CPU target share values.

If the CPU target shares for A and B are 60% and 40% respectively, then the CPU utilization for A and B stabilize at 60% and 40% respectively.

A third class, C, is added. This class is a group of CPU-bound jobs and should run with about half (or more) of the available CPU. Class C has limits of [20% - 100%] and CPU target shares of 100%. If C is in the same tier as A and B, then when C is starting, A and B see their CPU allocation decrease steeply and the three classes stabilize at 30%, 20% and 50%, respectively. Their targets in this case are also the minimum for A and B.

A system administrator might not want batch jobs to consume 50% of the CPU when other jobs, possibly of higher priority, are also running. In a situation like the previous example, C is placed in a lower priority

tier. C then receives whatever CPU remains after A and B receive their needs. In the above example, C receives no CPU time, because A and B are each capable of absorbing 100% of the CPU. In most situations, however, A and B, in a high-priority tier, are composed of interactive or transaction-oriented jobs, which do not use all of the CPU all of the time. C then receives some share of the CPU, for which it competes with other classes in the same or lower tiers.

Example of Workload Manager classes with memory limits

This example examines memory allocation to groups of processes with varying memory targets.

Three groups of processes must run: a group of interactive processes that need to run whenever they are used (PEOPLE), a batch job that always runs in the background (BATCH1), and a second, more important batch job, that runs every night (BATCH0).

PEOPLE has a specified memory minimum of 20%, a memory target of 50 shares, and a class tier value of 1. A 20% minimum limit ensures that the desktop applications in this class resume fairly quickly when users touch their keyboards.

BATCH1 has a memory minimum of 50%, a memory target of 50 shares, and a tier value of 3.

BATCH0 has a memory minimum of 80%, a memory target of 50 shares, and a tier value of 2.

Classes PEOPLE and BATCH1 have a total memory minimum limit of 70. Under normal operation (when BATCH0 is not running), both of these classes are allowed to get all of their reserved memory. They share the rest of the memory in the machine about half and half, even though they are in different tiers. At midnight when BATCH0 is started, the memory minimum total reaches 150. WLM ignores the minimum requirements for the lowest tiers until processes in the upper tiers exit. BATCH0 takes memory from the BATCH1 50% reserve, but not from the PEOPLE 20% reserve. After BATCH0 is done, memory reserves for tier 3 processes are honored again and the system returns to its normal memory balance.

Workload Manager commands

WLM offers commands that allow system administrators to perform a variety of functions.

These include:

- Create, modify, and delete super classes and subclasses, using the **mkclass**, **chclass**, and **rmclass** commands. These commands update the file's *classes*, *shares* and *limits*.
- Start, stop, and update WLM, using the **wlmcntrl** command.
- Check the WLM property files for a given configuration and determine to which class (superclass and subclass) a process with a given set of attributes is assigned using the **wlmcheck** command.
- Monitor the per-class resource utilization using the **wlmstat** (ASCII) command. Most of the performance tools, such as those started by the **svmon** and **topas** commands, have extensions to take into account the WLM classes and provide per-class and per-tier statistics using new command-line options.
- Flags in the **ps** command allow the user to display which class a process and its application tag is in. The **ps** command also allows the user to list all the processes belonging to a given superclass or subclass.
- There is no command line interface to manage the assignment rules. You must use the SMIT administration tool, or a text editor.

Device nodes

Devices are organized into clusters known as *nodes*. Each node is a logical subsystem of devices, where lower-level devices have a dependency on upper-level devices in child-parent relationships.

For example, the system node is the highest of all nodes and consists of all the physical devices in the system. The system device is the top of the node and below that are the bus and adapters that have a dependency on the higher-level system device. At the bottom of the hierarchy are devices to which

no other devices are connected. These devices have dependencies on all devices above them in the hierarchy.

At startup time, parent-child dependencies are used to configure all devices that make up a node. Configuration occurs from the top node down and any device having a dependency on a higher-level device is not configured until the higher-level device is configured.

The AIX operating system supports the multiple path I/O (MPIO) feature. If a device has an MPIO-capable device driver, it can have more than one parent within this hierarchy, which allows multiple, simultaneous communication paths between the device and a given machine or logical partition within a machine.

Device classes

Managing devices requires the operating system to comprehend what device connections are allowed. The operating system classifies devices hierarchically into three groups.

These groups are:

- Functional classes
- Functional subclasses
- Device types

Functional classes consist of devices that perform the same function. Printers, for example, comprise a functional class. Functional classes are grouped into subclasses according to certain device similarities. For example, printers have a serial or parallel interface. Serial printers are one subclass and parallel printers are another. Device types are classified according to their model and manufacturer.

Device classes define valid parent-child connections for the operating system. The hierarchy defines the possible subclasses that can be connected for each of the possible child connection locations. For example, the term RS-232 8-port adapter specifies that only devices belonging to the RS-232 subclass can be connected to any of the eight ports of the adapter.

Device classes and their hierarchical dependencies are maintained in an Object Data Manager (ODM) Device Configuration database.

Device configuration database and device management

Device information is contained in a predefined database or a customized database that makes up the device configuration database.

The predefined database contains configuration data for all possible devices supported by the system. The hierarchical device class information is contained in this database. The customized database contains configuration data for all currently defined and configured devices in the system. A record is kept of each device currently connected to your system.

The Configuration Manager is a program that automatically configures devices on your system during system startup and run time. The Configuration Manager uses the information from the predefined and customized databases during this process, and updates the customized database afterwards.

The multiple path I/O function uses a unique device identifier (UDID) to identify each MPIO-capable device, regardless of the path on which it was discovered. The UDID is saved in the device configuration database. When a device is discovered, the UDIDs in the database are checked to determine whether the device is new or whether the discovery is another path to an existing device. When multiple paths to a device are detected, the device driver or the Path Control Manager kernel extension decides which path to use for a particular request.

You can use the SMIT, or operating system commands to perform device management tasks such as deleting, adding, or configuring a device.

Device states

Devices that are connected to the system can be in one of four states.

Devices that are connected to the system can be in one of the following states:

Item	Description
Undefined	The device is unknown to the system.
Defined	Specific information about the device is recorded in the customized database, but it is unavailable to the system.
Available	A defined device is coupled to the operating system, or the defined device is configured.
Stopped	The device is unavailable but remains known by its device driver.

If a tty device and a printer alternately use the same tty connector, both a tty device and a printer are defined on the same parent and port in the device configuration database. Only one of these devices can be configured at a time. When the tty connector is configured, the printer specific setup information is retained until it is configured again. The device is not removed; it is in the defined state. Maintaining a device in defined state retains customized information for a device that is not currently in use, either before it is first made available or while it is temporarily removed from the system.

If a device driver exists for a device, the device can be made available through the device driver.

Some devices, in particular TCP/IP pseudo-devices, need the stopped state.

Device location codes

The *location code* is a path from the CPU drawer or system unit through the adapter, signal cables, and the asynchronous distribution box (if there is one) to the device or workstation. This code is another way of identifying physical devices.

The location code consists of up to four fields of information depending on the type of device. These fields represent drawer, slot, connector, and port. Each of these fields consists of two characters.

The location code of a drawer consists of only the drawer field and is simply a two-character code. The location code of an adapter consists of the drawer and slot fields and has the format *AA-BB*, where *AA* corresponds to the drawer location and *BB* indicates the bus and slot that contains the adapter. Other devices have location codes of formats *AA-BB-CC* or *AA-BB-CC-DD*, where *AA-BB* is the location code of the adapter to which the device is connected, *CC* corresponds to the connector on the adapter to which the device is connected, and *DD* corresponds to a port number or SCSI device address.

Adapter location codes

The location code for an adapter consists of two pairs of digits with the format *AA-BB*, where *AA* identifies the location code of the drawer containing the adapter and *BB* identifies the I/O bus and slot containing the card.

A value of 00 for the **AA** field means that the adapter is located in the CPU drawer or system unit, depending on the type of system. Any other value for the **AA** field indicates that the card is located in an I/O expansion drawer. In this case, the *AA* value identifies the I/O bus and slot number in the CPU drawer that contains the asynchronous expansion adapter. The first digit identifies the I/O bus with 0 corresponding to the standard I/O bus and 1 corresponding to the optional I/O bus. The second digit identifies the slot number on the indicated I/O bus.

The first digit of the **BB** field identifies the I/O board containing the adapter card. If the card is in the CPU drawer or system unit, this digit is 0 for the standard I/O bus and 1 for the optional I/O bus. If the card is in an I/O expansion drawer, this digit is 0. The second digit identifies the slot number on the indicated I/O bus (or slot number in the I/O expansion drawer) that contains the card.

A location code of 00-00 is used to identify the standard I/O board.

Examples:

Item	Description
00-05	Identifies an adapter card in slot 5 of the standard I/O board and is located in either the CPU drawer or the system unit, depending on the type of system.
00-12	Identifies an adapter in slot 2 of the optional I/O bus and is located in the CPU drawer.
18-05	Identifies an adapter card located in slot 5 of an I/O expansion drawer. The drawer is connected to the asynchronous expansion adapter located in slot 8 of the optional I/O bus in the CPU drawer.

Printer and plotter location codes

Location codes of 00-00-S1-00 or 00-00-S2-00 indicate the printer, plotter, or tty device is connected to the standard I/O board serial ports s1 or s2. A location code of 00-00-0P-00 indicates the parallel printer is connected to the standard I/O board parallel port.

Any other location code indicates that the printer, plotter, or tty device is connected to an adapter card other than the standard I/O board. For these printers, plotters, and tty devices, the location code format is *AA-BB-CC-DD*, where *AA-BB* indicates the location code of the controlling adapter.

Item	Description
AA	A value of 00 for the AA field indicates the adapter card is located in the CPU drawer or system unit depending on the type of system. Any other value for the AA field indicates the card is located in an I/O expansion drawer; in which case, the first digit identifies the I/O bus and the second digit identifies the slot number on the bus, in the CPU drawer, that contains the asynchronous expansion adapter to which the I/O expansion drawer is connected.
BB	The first digit of the BB field identifies the I/O bus containing the adapter card. If the card is in the CPU drawer or system unit, this digit is 0 for the standard I/O bus and 1 for the optional I/O bus. If the card is in an I/O expansion drawer, this digit is 0. The second digit identifies the slot number on the I/O bus (or slot number in the I/O expansion drawer) that contains the card.
CC	The CC field identifies the connector on the adapter card where the asynchronous distribution box is connected. Possible values are 01, 02, 03, and 04.
DD	The DD field identifies the port number on the asynchronous distribution box where the printer, plotter, or tty device is attached.

tty location codes

Location codes of 00-00-S1-00 or 00-00-S2-00 indicate the tty device is connected to the standard I/O serial ports s1 or s2.

Any other location code indicates the tty device is connected to an adapter card other than the standard I/O board. For these devices, the location code format is *AA-BB-CC-DD*, where *AA-BB* indicates the location code of the controlling adapter card.

Item	Description
AA	A value of 00 for the AA field indicates the adapter card is located in the CPU drawer or system unit depending on the type of system. Any other value for the AA field indicates the card is located in an I/O expansion drawer. In this case, the first digit identifies the I/O bus, and the second digit identifies the slot number on the bus in the CPU drawer that contains the asynchronous expansion adapter where the I/O expansion drawer is connected.

Item	Description
BB	The first digit of the BB field identifies the I/O bus containing the adapter card. If the card is in the CPU drawer or system unit, this digit will be 0 for the standard I/O bus and 1 for the optional I/O bus. If the card is in an I/O expansion drawer this digit is 0. The second digit identifies the slot number on the I/O bus (or slot number in the I/O expansion drawer) that contains the card.
CC	The CC field identifies the connector on the adapter card where the asynchronous distribution box is connected. Possible values are 01, 02, 03, and 04.
DD	The DD field identifies the port number on the asynchronous distribution box where the tty device is attached.

SCSI device location codes

The following are location codes for SCSI devices.

These location codes apply to all SCSI devices including:

- CD-ROMs
- Disks
- Initiator devices
- Read/write optical drives
- Tapes
- Target mode

The location code format is AA-BB-CC-S, L. The **AA-BB** fields identify the location code of the SCSI adapter controlling the SCSI device.

Item	Description
AA	A value of 00 for the AA field indicates the controlling adapter card is located in the CPU drawer or system unit, depending on the type of system.
BB	The BB field identifies the I/O bus and slot containing the card. The first digit identifies the I/O bus. It is 0 for the standard I/O bus and 1 for the optional I/O bus. The second digit is the slot on the indicated I/O bus containing the card. A value of 00 for the BB field indicates the standard SCSI controller.
CC	The CC field identifies the SCSI bus of the card that the device is attached to. For a card that provides only a single SCSI bus, this field is set to 00. Otherwise, a value of 00 indicates a device attached to the internal SCSI bus of the card, and a value of 01 indicates a device attached to the external SCSI bus of the card.
S,L	The S,L field identifies the SCSI ID and logical unit number (LUN) of the SCSI device. The S value indicates the SCSI ID and the L value indicates the LUN.

Dials/LPFKeys location codes

For a Dials/LPFKeys device attached to a graphics input adapter, the location code format is AA-BB-CC.

The individual fields are interpreted as follows:

Item	Description
AA	A value of 00 for the AA field indicates the controlling adapter card is located in the CPU drawer or system unit, depending on the type of system.

Item	Description
-------------	--------------------

- | | |
|-----------|---|
| BB | The BB field identifies the I/O bus and slot containing the card. The first digit identifies the I/O bus. It is 0 for the standard I/O bus and 1 for the optional I/O bus. The second digit is the slot on the indicated I/O bus that contains the card. |
| CC | The CC field indicates the card connector where the device is attached. The value is either 01 or 02, depending on whether the attached device is port 1 or port 2 on the card. |

Note: Serially-attached Dials/LPFKeys devices do not indicate location codes. This is because these devices are considered to be attached to a tty device. The tty device is specified by the user during Dials/LPFKeys definition.

Multiprotocol port location codes

The location code for a multiprotocol port is of the format *AA-BB-CC-DD* where *AA-BB* indicates the location code of the multiprotocol adapter card.

The individual fields are interpreted as follows:

Item	Description
-------------	--------------------

- | | |
|-----------|---|
| AA | A value of 00 for the AA field indicates the multiprotocol adapter card is located in the CPU drawer or system unit, depending on the type of system. |
| BB | The BB field identifies the I/O bus and slot containing the card. The first digit identifies the I/O bus. It is 0 for the standard I/O bus and 1 for the optional I/O bus. The second digit is the slot on the indicated I/O bus that contains the card. |
| CC | The CC field identifies the connector on the adapter card to which the multiprotocol distribution box is connected. The value is always 01. |
| DD | The DD field identifies the physical port number on the multiprotocol distribution box. Possible values are 00, 01, 02, and 03. |

AIX device drivers

Many computer programs are dedicated to work with devices that are attached to the computer in some way. For example, some programs send control characters to a printer, some programs receive characters from a console, and some programs read data from a tape. Each of these programs is a *device driver* program because the program is dedicated to handle input from or output to a device. Such programs are a part of, or an extension of, the computer's operating system.

Any operating system that supports multitasking, such as the AIX operating system, needs a mechanism to prevent one program from writing data to, or changing the state of, a device that is already being accessed by another program. So, a multitasking operating system depends on the computer's processors to distinguish between privileged and non-privileged execution of instructions. Therefore, you must distinguish between programs that run in privileged mode (kernel mode) and programs that run in user mode. The AIX kernel consists of all software programs that run in kernel mode.

Some AIX programs that run in user mode, such as a network adapter or a device that is attached to a USB port, can access device driver programs. However, these AIX programs can access the device driver programs only by using software that is part of the kernel. Because kernel device drivers are more complex than the device drivers that run in user mode, the device driver term refers to the software that controls a device when the program is running in kernel mode.

The routines for the device driver programs are written in C and compiled to produce one or two Extended Object File Format (XCOFF or XCOFF64) object files. Beginning with AIX Version 6.1, all device driver programs are 64-bit (XCOFF64). The 32-bit device driver programs are not supported. The object files are linked to enable the kernel loader to resolve the kernel symbols. When the object files are linked, the

kernel loader section is updated with a list of symbols to import from the kernel. The kernel symbols are in the `/lib/kernex.exp` file. This linking also establishes the configuration routine of the device driver as the default entry point for beginning execution.

For more information about AIX device driver programs, see [Writing an AIX Device Driver Program](#). This document explains the following items:

- Overview of AIX device drivers, entry points, and types
- Programmed I/O operations and Direct Memory Access (DMA) for PCI adapters
- Interrupts and interrupt handling
- Memory management
- Serialization and synchronization, locking, and timers
- Device configuration methods

For information about ODM for PCI/PCIe adapters, see [Customizing AIX ODM for PCI/PCIe adapters](#).

Setting up an iSCSI offload adapter

Setting up the iSCSI offload adapter involves configuring the adapter and adding or updating targets. These instructions apply only to the iSCSI offload adapter. For information about configuring the iSCSI software initiator, refer to [iSCSI software initiator and software target](#).

Configuring the iSCSI adapter in AIX

iSCSI adapter configuration is a very simple and straightforward task.

1. Enter **smit iscsi** at the AIX command prompt.
The iSCSI screen displays.
2. Select **iSCSI Adapter** from the iSCSI screen.
The iSCSI Adapter screen displays.
3. Select **Change / Show Characteristics of an iSCSI Adapter** from the iSCSI Adapter screen.
The Change / Show Characteristics of an iSCSI Adapter screen displays.
4. Select the iSCSI adapter that you want to configure from the list.
A configuration screen displays, similar to the following example.

	[Entry Fields]
iSCSI Adapter	ics0
Description	iSCSI Adapter
Status	Available
Location	10-60
Max. number of commands to queue to adapter	[200] + +#
Maximum Transfer Size	[0x100000] +
Discovery Filename	[/etc/iscsi/targetshw]
Discovery Policy	[file]
Automatic Discovery Secrets Filename	[/etc/iscsi/autosecret]
Adapter IP Address	[10.1.4.187]
Adapter Subnet Mask	[255.255.255.0]
Adapter Gateway Address	[10.1.4.1]
Apply change to DATABASE only	no +

Note: If you have a question about the purpose of a particular field, place the cursor in the field and press **F1** for help.

To use flat file discovery, type **file** in the **Discovery Policy** field. To use ODM discovery, type **odm** in the **Discovery Policy** field. For DHCP-discovered iSCSI targets, type **slp** in the **Discovery Policy** field.

Updating the flat file of an iSCSI target

The flat file is the static configuration file used to configure iSCSI targets. Its default filename is `/etc/iscsi/targetshw`.

You must explicitly specify all relevant iSCSI target discovery properties in the flat file.

Related information

[targets File](#)

Adding a statically-discovered iSCSI target into ODM

When using ODM discovery policy for either the iSCSI adapter or the iSCSI software initiator, the iSCSI driver obtains the iSCSI target descriptions from ODM.

You can use AIX commands or SMIT to manipulate iSCSI target information in ODM. The **chiscsi**, **lsiscsi**, **mkiscsi**, and **rmiscsi** commands change, display, add, and remove iSCSI target information from ODM.

To add one statically-discovered iSCSI target into ODM using SMIT, do the following:

1. Enter **smit iscsi** at the AIX command prompt.
The iSCSI screen displays.
2. Select **iSCSI Target Device Parameters in ODM** from the iSCSI screen.
The iSCSI Target Device Parameters in ODM screen displays.
3. Select **Add an iSCSI Target Device in ODM** from the iSCSI screen.
The Add an iSCSI Target Device in ODM screen displays.
4. Select the iSCSI device that you want to configure from the list. The list might include both iSCSI offload adapters (devices starting with "ics") and the iSCSI software initiator (the device starting with "iscsi").
5. Select the iSCSI device that you want to configure from the list. The list might include both iSCSI offload adapters (devices starting with "ics") and the iSCSI software initiator (the device starting with "iscsi").
6. Enter the appropriate information in the fields.
Below is an example.

```
[Entry Fields]
iscsi Adapter          ics0
iscsi Target Name      [iqn.com.ibm.tgt:0]
IP Address of iSCSI Target [10.1.4.25]
Port Number of iSCSI Target [3260]
Password               [my password]
User Name               [user name]
```

Note: If you have a question about the purpose of a particular field, place the cursor in the field and press **F1** for help.

Adding statically-discovered iSCSI targets from a flat file into ODM

You can use SMIT to import a flat file's information into ODM.

1. Enter **smit iscsi** at the AIX command prompt.
The iSCSI screen displays.
2. Select **iSCSI Target Device Parameters in ODM** from the iSCSI screen.
The iSCSI Target Device Parameters in ODM screen displays.
3. Select **Add an iSCSI Target Device in ODM** from the iSCSI screen.
The Add an iSCSI Target Device in ODM screen displays.
4. Select **Add iSCSI Target Device Data in ODM from a File** from the iSCSI screen.
The Add iSCSI Target Device Data in ODM from a File screen displays.
5. Select the iSCSI adapter that you want to configure from the list.

The Add iSCSI Target Device Data in ODM from a File screen for the iSCSI adapter that you selected displays.

6. Enter the appropriate information in the fields.
Below is an example.

[Entry Fields]			
iSCSI Protocol Device	iscsi3		
iSCSI Group	[static]	+	
Filename of iSCSI Targets	[/etc/iscsi/targetshw]		+

Note: If you have a question about the purpose of a particular field, place the cursor in the field and press **F1** for help.

PCI hot plug management

You can insert a new PCI hot plug adapter into an available PCI slot while the operating system is running.

This can be another adapter of the same type that is currently installed or of a different type of PCI adapter. New resources are made available to the operating system and applications without having to restart the operating system. Some reasons for adding a hot plug adapter might include:

- Adding additional function or capacity to your existing hardware and firmware.
- Migrating PCI adapters from a system that no longer requires the function provided by those adapters.
- Installing a new system for which adapter cards become available after the initial configuration of optional hardware subsystems, including PCI adapters, and the installation and start of the operating system.

Note: If you add an adapter using a PCI hot plug replace or add operation, or using Dynamic Logical Partitioning, it and its child devices may not be available for specification as a boot device using the **bootlist** command. You may have to restart the machine to make all potential boot devices known to the operating system. An adapter already listed in the boot list, which is then replaced by the exact type of adapter, is still a valid boot device.

You can also remove a defective or failing PCI hot plug adapter or exchange it with another of the same type without shutting down or powering off the system. When you exchange the adapter, the existing device driver supports the adapter because it is of the same type. Device configuration and configuration information about devices below the adapter are retained for the replacement device. Some reasons for replacing an adapter might include:

- Temporarily replacing the card to aid in determining a problem or to isolate a failing FRU.
- Replacing a flawed, failing, or intermittently failing adapter with a functional card.
- Replacing a failing redundant adapter in an HACMP or multipath I/O configuration.

When you remove a hot plug adapter, the resources provided by the adapter become unavailable to the operating system and applications. Some reasons for removing an adapter might include:

- Removing existing I/O subsystems.
- Removing an adapter that is no longer required or is failing and a replacement card is not available.
- Migrating an adapter to another system when the function is no longer required on the system from which it is being removed.

Before you can remove or replace a hot plug device, it must be unconfigured. The associated device driver must free any system resources that it has allocated for the device. This includes unpinning and freeing memory, undefining interrupt and EPOW handlers, releasing DMA and timer resources, and any other required steps. The driver must also ensure that interrupts, bus memory, and bus I/O are disabled on the device.

The system administrator must perform the following tasks before and after removing an adapter:

- Terminate and restore applications, daemons, or processes using the device.

- Unmount and remount file systems.
- Remove and recreate device definitions and perform other operations necessary to free up a device in use.
- Put the system into a safe state to be serviced.
- Obtain and install any required device drivers.

The remove and replace operations fail unless the device connected to the identified slot has been unconfigured and is in the defined state. You can do this with the **rmdev** command. Before placing the adapter in the defined state, close all applications that are using the adapter, otherwise, the command will be unsuccessful. For more information about the **rmdev** command, see [rmdev](#).

In some cases, the you can also perform the following tasks:

- Prepare a PCI hot plug adapter to be inserted, removed, or replaced.
- Identify slots or PCI adapters that are involved in the hot plug operation.
- Remove or insert PCI hot plug adapters.

Note: During the hot plug operations, the Object Data Manager (ODM) remains locked. Hence, the other tasks that require the ODM might hang or fail. The cluster-wide configuration changes that are initiated by other nodes might also hang or fail in a cluster. Therefore, ensure that such tasks are not performed until the hot plug operation is complete.



Attention: Although PCI hot plug management provides the capability of adding, removing, and replacing PCI adapters without powering off the system or restarting the operating system, not all devices in hot plug slots can be managed in this fashion. For example, the hard disk that makes up the rootvg volume group or the I/O controller to which it is attached cannot be removed or replaced without powering off the system because it is necessary for running the operating system. If the rootvg volume group is mirrored, you can use the **chpv** command to take the disks offline. If the rootvg volume group resides on one or more disks that are Multi-Path I/O (MPIO) enabled and connected to multiple I/O controllers, one of these I/O controllers can be removed (or replaced) without rebooting the system. In this situation, all paths from the I/O controller to be removed (or replaced) should be unconfigured using the **rmdev -R** command on the adapter. This will unconfigure the paths and the adapter. You can then proceed with Hot Plug Management. Before you attempt to remove or insert PCI hot plug adapters, refer to the *PCI Adapter Placement Reference*, (shipped with system units that support hot plug), to determine whether your adapter can be hot-swapped. Refer to your system unit documentation for instructions for installing or removing adapters.

Displaying PCI hot-plug slot information

Before you add, remove, or replace a hot-plug adapter, you can display information about the PCI hot-plug slots in a machine.

You can display the following information:

- A list of all the PCI hot-plug slots in the machine
- Whether a slot is available or empty
- Slots that are currently in use
- The characteristics of a specific slot such as slot name, description, connector type, and the attached device name

You can use SMIT or system commands. To perform these tasks, you must log in as root user.

SMIT fastpath procedure

1. Type `smit devdpc` at the system prompt, then press Enter.
2. Use the SMIT dialogs to complete the task.

To obtain additional information for completing the task, you can select the F1 Help key in the SMIT dialogs.

Commands procedure

You can use the following commands to display information about hot-plug slots and connected devices:

- The `lsslot` command displays a list of all the PCI hot-plug slots and their characteristics.
- The `lsdev` command displays the current state of all the devices installed in your system.

Unconfiguring PCI communications adapters

The following is an overview of the process for unconfiguring PCI communications adapters. This includes Ethernet, Token-ring, FDDI, and ATM adapters.

If your application is using TCP/IP protocol, you must remove the TCP/IP interface for the adapter from the network interface list before you can place the adapter in the defined state. Use the `netstat` command to determine whether your adapter is configured for TCP/IP and to check the active network interfaces on your adapter. For information about the `netstat` command, see [netstat](#).

An Ethernet adapter can have two interfaces: Standard Ethernet (enX) or IEEE 802.3 (etX). X is the same number in the entX adapter name. Only one of these interfaces can be using TCP/IP at a time. For example, Ethernet adapter ent0 can have en0 and et0 interfaces.

A Token ring adapter can have only one interface: Token-ring (trX). X is the same number in the tokX adapter name. For example, Token-ring adapter tok0 has a tr0 interface.

An ATM adapter can have only one atm interface: ATM (atX). X is the same number in the atmX adapter name. For example, ATM adapter atm0 has an at0 interface. However, ATM adapters can have multiple emulated clients running over a single adapter.

The `ifconfig` command removes an interface from the network. The `rmdev` command unconfigures the PCI device while retaining its device definition in the Customized Devices Object Class. Once the adapter is in the defined state, you can use the `drslot` command to remove the adapter.

To unconfigure the children of PCI bus pci1 and all other devices under them while retaining their device definitions in the Customized Devices object class, type:

```
rmdev -p pci1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
scsi1 Defined
ent0 Defined
```

Removing or replacing a PCI hot plug adapter

You can remove or replace a PCI hot plug adapter from the system unit without shutting down the operating system or turning off the system power. Removing an adapter makes the resources provided by that adapter unavailable to the operating system and applications.

Before you can remove an adapter, you must unconfigure it.

The following are the procedures for removing a PCI hot plug adapter. You can complete these tasks with the SMIT or system commands. To perform these tasks, you must log in as root user.

Replacing an adapter with another adapter of the same type retains the replaced adapter's configuration information and compares the information to the card that replaces it. The existing device driver of the replaced adapter must be able to support the replacement adapter.

SMIT fastpath procedure

1. Type `smit devdrcpci` at the system prompt, then press Enter.
2. Use the SMIT dialogs to complete the task.

To obtain additional information for completing the task, you can select the F1 Help key in the SMIT dialogs.

Commands procedure

You can use the following commands to display information about hot plug slots and connected devices and to remove a PCI hot plug adapter:

- The **lsslot** command displays a list of all the PCI hot plug slots and their characteristics.
- The **lsdev** command displays the current state of all the devices installed in your system.
- The **drslot** command prepares a hot plug slot for removal of a hot plug adapter.

Adding a PCI hot plug adapter

You can add a PCI hot plug adapter into an available slot in the system unit and make new resources available to the operating system and applications without having to reboot the operating system. The adapter can be another adapter type that is currently installed or it can be a different adapter type.

The following are the procedures for adding a new PCI hot plug adapter.



Attention: Before you attempt to add PCI hot plug adapters, refer to the *PCI Adapter Placement Reference*, shipped with system units that support hot plug, to determine whether your adapter can be hot plugged. Refer to your system unit documentation for instructions for installing or removing adapters.

Adding a new PCI hot plug adapter involves the following tasks:

- Finding and identifying an available slot in the machine
- Preparing the slot for configuring the adapter
- Installing the device driver, if necessary
- Configuring the new adapter

You can use SMIT or system commands. To perform these tasks, you must log in as root user.

Note: When you add a hot plug adapter to the system, that adapter and its child devices might not be available for specification as a boot device using the **bootlist** command. You might be required to reboot your system to make all potential boot devices known to the operating system.

SMIT fastpath procedure

1. Type `smit devdpci` at the system prompt, then press Enter.
2. Use the SMIT dialogs to complete the task.

To obtain additional information for completing the task, you can select the F1 Help key in the SMIT dialogs.

Commands procedure

You can use the following commands to display information about PCI hot plug slots and connected devices and to add a PCI hot plug adapter:

- The **lsslot** command displays a list of all the hot plug slots and their characteristics.
- The **lsdev** command displays the current state of all the devices installed in your system.
- The **drslot** command prepares a hot plug slot for adding or removing a hot plug adapter.

Multiple Path I/O

With Multiple Path I/O (MPIO), a device can be uniquely detected through one or more physical connections, or *paths*.

A path-control module (PCM) provides the path management functions.

An MPIO-capable device driver can control more than one type of target device. A PCM can support one or more specific devices. Therefore, one device driver can be interfaced to multiple PCMs that control the I/O across the paths to each of the target devices.

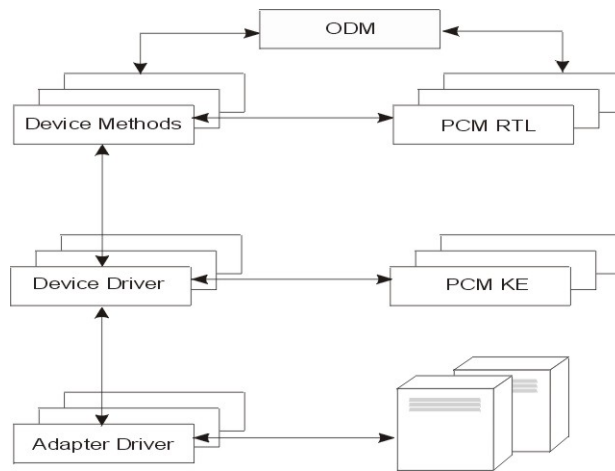


Figure 13. MPIIO Component Interaction

Before a device can take advantage of MPIIO, the device's driver, methods, and predefined attributes in the Object Data Manager (ODM) must be modified to support detection, configuration, and management of multiple paths. The parallel SCSI and Fibre Channel disk device drivers and their device methods support MPIIO disk devices. The iSCSI disk devices are supported as MPIIO devices. The Fibre Channel tape device driver and its device methods support MPIIO tape devices. Also, the predefined attributes for some devices in the ODM have been modified for MPIIO.

The AIX PCM consists of: the PCM RTL configuration module and the PCM KE kernel extension. The PCM RTL is a run-time loadable module that enables the device methods to detect additional PCM KE device-specific or path ODM attributes that the PCM KE requires. The PCM RTL is loaded by a device method. One or more routines within the PCM RTL are then accessed to perform specific operations that initialize or modify PM KE variables.

The PCM KE supplies path-control management capabilities to any device driver that supports the MPIIO interface. The PCM KE depends on device configuration to detect paths and communicate that information to the device driver. Each MPIIO-capable device driver adds the paths to a device from its immediate parent or parents. The maintenance and scheduling of I/O across different paths is provided by the PCM KE and is not apparent to the MPIIO-capable device driver.

The PCM KE can provide more than one routing algorithm, which can be selected by the user. The PCM KE also helps collect information that can be used to determine and select the best path for any I/O request. The PCM KE can select the best path based on a variety of criteria, including load balancing, connection speed, connection failure, and so on.

The AIX PCM has a health-check capability that can be used to do the following:

- Check the paths and determine which paths are currently usable for sending I/O
- Enable a path that was previously marked failed because of a temporary path fault (for example, when a cable to a device was removed and then reconnected)
- Check currently unused paths that would be used if a failover occurred (for example, when the algorithm attribute value is `failover`, the health check can test the alternate paths)

Not all disk devices and tape devices can be detected and configured using the AIX default PCMs. The AIX default PCMs consist of two path control modules, one to manage disk devices and another to manage tape devices. If your device is not detected, check with the device vendor to determine if a PCM is available for your device.

MPIO-capable device management

The Multiple Path I/O (MPIIO) feature can be used to define alternate paths to a device for failover purposes.

Failover is a path-management algorithm that improves the reliability and availability of a device because the system automatically detects when one I/O path fails and re-routes I/O through an alternate path.

All SCSI SCSD disk drives are automatically configured as MPIO devices and a select number of Fibre Channel disk drives can be configured as MPIO Other disk. Other devices can be supported, providing the device driver is compatible with the MPIO implementation in AIX.

MPIO is installed and configured as part of BOS installation. No further configuration is required, but you can add, remove, re-configure, enable, and disable devices (or device paths) using SMIT, or the command-line interface. The following commands help manage MPIO paths:

mkpath

Adds a path to a target device.

rmpath

Removes a path to a target device.

chpath

Changes an attribute or the operational status of a path to a target device.

lsmPIO

Displays information such as the status, configuration, and statistics about multiPath I/O (MPIO) storage devices.

lspath

Displays information about paths to a target device.

If you are performing a BOS installation or a mksysb installation on a disk with multiple ports such as Multipath I/O (MPIO), the port must remain active for the duration of the install.

Cabling a SCSI device as an MPIO device

A SCSI device can be supported by a maximum of two adapters when configured as a MPIO-capable device.

To cable a parallel SCSI device as an MPIO device, use the following simple configuration as an example. The following is the minimum configuration that must be done; your device might require additional configuration.

1. With the power off, install two SCSI adapters.
2. Cable the device to both SCSI adapters.
3. Power on the system.
4. Change the settings on one of the adapters to a unique SCSI ID. By default, SCSI adapters have a SCSI ID of 7. Because each ID must be unique, change one adapter to another number, for example, 6.
5. Run the **cfgmgr** command.
6. To verify the configuration, type the following on the command line:

```
lspath -l hdiskX
```

where X is the logical number of the newly configured device. The command output should display two paths and their status.

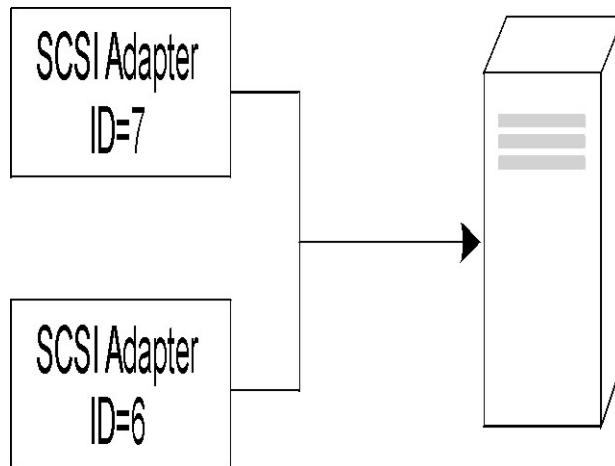


Figure 14. Cable Configuration for MPIIO SCSI Device

This illustration shows cabling two SCSI adapters to the same device.

Cabling a Fibre Channel device as an MPIIO device

A Fibre Channel device can be cabled to multiple adapters. There is no limit within the software.

To cable a Fibre Channel device as an MPIIO device, use the following simple configuration as an example. The following is the minimum configuration that must be done; your device might require additional configuration.

1. With the power off, install two Fibre Channel adapters.
2. Cable the adapters to a switch or hub.
3. Cable the device to the switch or hub.
4. Power on the system.
5. To verify the configuration, type the following on the command line:

```
lspath -l hdiskX
```

where X is the logical number of the newly configured device. The command output should display one path for each adapter you installed and the status of each.

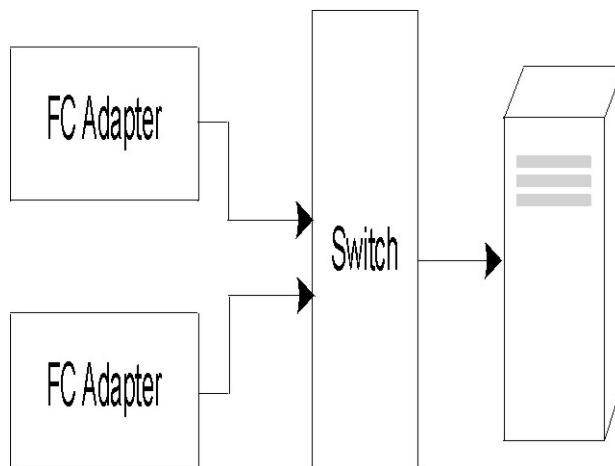


Figure 15. Cable Configuration for MPIIO Fibre Channel Device

Configuring an MPIO device

Configuring an MPIO-capable device uses the same commands as a non-MPIO device.

The **cfgmgr**, **mkdev**, **chdev**, **rmdev** and **lsdev** commands support managing MPIO device instances and display their attributes. An MPIO device instance also has path instances associated with the device instance. The **mkpath**, **chpath**, **rmpath**, and **lspath** commands manage path instances and display their attributes.

A path instance can be added or removed from an MPIO device without the device being unconfigured.

An MPIO device can have additional attributes, but the required attributes that all MPIO devices must support are `reserve_policy` and `algorithm`. The `reserve_policy` attribute determines the type of reserve methodology the device driver will implement when the device is opened, and it can be used to limit device access from other adapters, whether on the same system or another system. The `algorithm` attribute defines the methodology that the PCM uses to manage I/O across the paths configured for a device.

Encrypting physical volumes

Starting from IBM® AIX 7.3 with Technology Level 1, you can encrypt the physical volumes that uses the small computer systems Interconnect® (SCSI) protocol. Data that is written to the physical volume is encrypted and data that is read from the physical volume is decrypted during an I/O operation. Data encryption on a physical volume is not enabled by default. To encrypt data that is stored in a physical volume, you must enable encryption for the physical volume. When encryption is enabled for a physical volume, all previous data on that physical volume is deleted. The size of the encrypted physical volume is smaller than the size of the physical volume before encryption because the encryption feature reserves some space on the physical volume for the encryption process.

Each physical volume is encrypted by using a unique key, that is known as a data encryption key. The data encryption key is required for encrypting and decrypting all data that is stored in the physical volume. Additional user keys are used to encrypt the data encryption key. You can change the user key without changing the data encryption key. The user key can be a typed passphrase, or it can be stored in a platform keystore (PKS) or in a network key manager.

The **hdcryptmgr** command manages encrypted physical volumes and encrypted logical volumes. The **hdcryptmgr** command provides action parameters that are specific to physical volumes such as **pvenable**, **pvdisable**, and **showpv** action parameters. Action parameters, such as **authadd**, **authcheck**, and **authdelete** can be used on both physical volumes and logical volumes.

To enable encryption on a physical volume (hdisk10), enter the following command:

```
hdcryptmgr pvenable hdisk10
```

The **pvenable** action parameter of the **hdcryptmgr** command prompts you to confirm whether data on the physical volume (hdisk10) can be deleted. The **pvenable** action parameter also prompts you to provide the passphrase to use to encrypt the data encryption key. The **pvenable** action parameter automatically runs the **authinit** action parameter to enable the physical volume for encryption and to initialize the first authentication method. Also user keys can be added by using the **authadd** action parameter. Any data that is written to the physical volume is encrypted and data that is read from the physical volume is decrypted after the **pvenable** action parameter runs successfully.

Encrypted physical volumes can only be shared with other L PARS that are running the AIX 7.3 Technology Level 1 or later. Sharing encrypted physical volumes with older levels of AIX might cause data corruption. If encryption of a physical volume is enabled or disabled by using the **pvenable** or **pvdisable** action parameters on one LPAR, you must run the **rmdev** and **mkdev** command for the shared physical volume on the other LPARS or reboot the other LPARs to recognize the changes to the encryption state of the shared physical volume.

When the AIX operating system is rebooted, the boot process identifies the encrypted disks or physical volumes. The encrypted disk is unlocked during the boot process if the disk uses an authentication method that supports automatic retrieval of the key. The disk remains in a locked state if the encrypted

disk supports only a typed passphrase or a key file method. If you perform I/O operations on an encrypted disk that is in locked state, the operation fails with a permission error. If the key storage method does not allow automatic unlocking, the **authunlock** action parameter must be used to unlock the encrypted physical volume to access the encrypted physical volume.

To manually unlock the encrypted disk, enter the following command:

```
hdcryptmgr authunlock diskname
```

The **authunlock** action parameter of the **hdcryptmgr** command prompts you to provide the passphrase to unlock the encrypted disk.

Disks that can be encrypted have the following new attributes:

encrypt_enabled

If this attribute is set to yes, it indicates that the disk is encrypted. If this attribute is set to no, it indicates that the disk is unencrypted.

encrypt_md_loc

Indicates the location of the encryption metadata for the encrypted disk. This attribute has the following values: **none** for a non-encrypted disc or **on_disk** for an encrypted disc.

Supported multi-path devices

The AIX default PCMs support a set of disk devices and tape devices defined in the `devices.common.IBM.mpio.rte` fileset.

Devices not supported by the AIX disk PCMs or tape PCMs require the device vendor to provide attributes predefined in the ODM, a PCM, and any other supporting code necessary to recognize the device as MPIO-capable.

To determine which disk devices are supported by the AIX disk PCM, run the following script :

```
odmget -qDvDr=aixdiskpcmke PdDv | grep uniquetype | while read line
do
    utype=`echo $line | cut -d'"' -f2`
    dvc=`odmget -q"uniquetype=$utype AND attribute=dvc_support" PdAt`
    echo $dvc | grep values | cut -d'"' -f2
done
```

To determine which tape devices are supported by the AIX disk PCM, run the following script :

```
odmget -qDvDr=aixtapepcmke PdDv | grep uniquetype | while read line
do
    utype=`echo $line | cut -d'"' -f2`
    dvc=`odmget -q"uniquetype=$utype AND attribute=dvc_support" PdAt`
    echo $dvc | grep values | cut -d'"' -f2
done
```

The script output displays a list of unique device types supported by the AIX default PCMs. The three device types supported by the AIX Disk PCM are *self-configuring parallel SCSI disk* (`disk/scsi/scsd`) and *MPIO other FC disk* (`disk/fcp/mpioosdisk`), and *MPIO other iSCSI* (`disk/iscsi/mpioosdisk`). The device type supported by the AIX tape PCM is *MPIO other FC tape* (`tape/fcp/mpioost`).

MPIO other FC disk and *MPIO other FC tape* devices are a subset of other Fibre Channel disks and other Fibre Channel tapes, respectively. A device is supported as an *MPIO other FC* device only if there are no vendor-provided ODM predefined attributes and the device has been certified to operate with one of the AIX default PCMs. Certification does not guarantee that all device capabilities are supported when the device is configured as an *MPIO other FC* device.

An *MPIO other iSCSI disk* is a subset of other iSCSI disks. A device is supported as an *MPIO other iSCSI disk* only if there is no vendor-provided ODM predefined attributes and the device has been certified to operate with the AIX PCM. Certification does not guarantee that all device capabilities are supported when the device is configured as an *MPIO other iSCSI disk*.

To determine which devices are supported as *MPIO other FC* disk, run the following script:

```
odmget -q uniqueness=disk/fcp/mpioosdisk PdAt | grep deflt | cut -d'"' -f2
```

To determine which devices are supported as *MPIO other FC* tape, run the following script:

```
odmget -q "uniquetype=tape/fcp/mpioosdisk AND attribute=mpio_model_map PdAt | grep deflt | cut -d'"' -f2
```

To determine which devices are supported as *MPIO other iSCSI* disks, run the following script:

```
odmget -q uniqueness=disk/iscsi/mpioosdisk PdAt | grep deflt | cut -d'"' -f2
```

The script output displays a list of inquiry data that contains the device vendor and model.

To display all MPIO-capable devices that are installed on the system, yes it would run the following script:

```
odmget -q attribute=unique_id PdAt | grep uniqueness | cut -d'"' -f2
```

The script output displays a list of unique MPIO-capable device types, supported by the AIX default PCMs and vendor provided PCMs.

MPIO device attributes

The following attributes are supported only by multipath devices. The attributes can be displayed or changed by using the SMIT, or commands (in particular, the **lsattr** and the **chdev** commands).

Some of the Multiple Path I/O (MPIO) device attributes are enabled for concurrent update of the attribute. You can update the attribute values while the disk is open and is in use, and the new values take effect immediately. For some attributes, particularly the **reserve_policy** attribute, there might be restrictions on when the attribute can be changed or what new values the attribute can accept. For example, if a disk is open and currently in use as a cluster repository disk, the AIX operating system blocks any attempt to set a reserve policy on the disk because it causes other cluster nodes to lose access to the repository.

The required device attributes that all MPIO devices must support is **reserve_policy**. Typically, a multipath device also has the **PR_key_value** device attribute. A multipath device can have additional device-specific attributes. Other device-specific attributes are as follows:

FC3_REC

Indicates whether the device must turn on error recovery that uses Fibre Channel class 3 or not. Enabling this feature improves error detection and error recovery for certain types of fabric errors that are related to Fibre Channel. This attribute is available only on a limited set of devices. This attribute can have the following values:

true

Enables error recovery that uses Fibre Channel class 3.

false

Disables error recovery that uses Fibre Channel class 3 error recovery.

reserve_policy

Defines whether a reservation methodology is employed when the device is opened. The values are as follows:

no_reserve

Does not apply a reservation methodology for the device. The device might be accessed by other initiators, and these initiators might be on other host systems.

single_path

Applies a SCSI2 reserve methodology for the device, which means the device can be accessed only by the initiator that issued the reserve. This policy prevents other initiators in the same host or on other hosts from accessing the device. This policy uses the SCSI2 reserve to lock the device to a single initiator (path), and commands routed through any other path result in a reservation conflict.

Path selection algorithms that alternate commands among multiple paths can result in thrashing when the `single_path` value is selected. As an example, assume a device-specific PCM has a required attribute that is set to a value that distributes I/O across multiple paths. When `single_path` is in effect, the disk driver must issue a bus device reset (BDR) and then issue a reserve using a new path for sending the next command to break the previous reservation. Each time a different path is selected, thrashing occurs and performance degrades because of the overhead of sending a BDR and issuing a reserve to the target device. (The AIX PCM does not allow you to select an algorithm that could cause thrashing.)

PR_exclusive

Applies a SCSI3 persistent-reserve, exclusive-host methodology when the device is opened. The `PR_key_value` attribute value must be unique for each host system. The `PR_key_value` attribute is used to prevent access to the device by initiators from other host systems.

PR_shared

Applies a SCSI3 persistent-reserve, shared-host methodology when the device is opened. The `PR_key_value` value must be a unique value for each host system. Initiators from other host systems are required to register before they can access the device.

PR_key_value

Required only if the device supports any of the persistent reserve policies (`PR_exclusive` or `PR_shared`).

rw_timeout and min_rw_to

The `rw_timeout` attribute specifies the time, in seconds, that a SCSI command is allowed for completion each time it is issued to a storage device. In an MPIO environment, a command can experience more than one timeout as it is retried on different paths.

The disk ODM, contained in AIX for Fibre Channel attached disks, uses a default value of 30 seconds for `rw_timeout`. For some devices, AIX allows setting the `rw_timeout` attribute to values as low as 10 seconds.

AIX uses a special unique type for some MPIO FC attached disks, `disk/fcp/mpioosdisk`. This unique type is used for various storage device models, which might have different requirements for the smallest allowed value of the `rw_timeout` attribute. As a result, disks of this type have a `min_rw_to` attribute that specifies the actual minimum timeout value for that particular disk.

For example, the `lsattr -R1 hdisk5 -a rw_timeout` command might show that `hdisk5` allows an `rw_timeout` value of 10 seconds. The value of the `min_rw_to` attribute, however, might provide a different, higher minimum value for the `rw_timeout` attribute. When you are using a storage device that is provided by a vendor other than IBM, obtain and install the ODM package from the storage vendor. The ODM package from the storage vendor might have a different range of values that are allowed for the `rw_timeout` attribute.

rw_max_time

This attribute specifies an approximate maximum time taken by an MPIO device, in seconds, to complete each I/O operation. The time is measured from when the I/O request is issued to the disk driver to when the disk driver returns the I/O request to the calling process. This attribute can be used when the disk is part of a redundant configuration, such as Logical Volume Manager (LVM) mirroring configuration. When this attribute is set, if an I/O operation encounters errors such as command timeout, the I/O operation might fail before completing all of the normal retry operations. Thus, this attribute provides quicker redundancy.

When this attribute is set to the default value of 0, the disk driver and Path Control Module (PCM) retry I/O operations until the threshold value of the retry counter is reached. The retry I/O operations might take several minutes depending on the number of errors and the configuration of the disk. If the `rw_max_time` attribute is set to a nonzero value and if you use a Fibre Channel adapter that provides the `io_thrshld_tmr` attribute, set the `io_thrshld_tmr` attribute to yes for best results.

io_thrshld_tmr

The `io_thrshld_tmr` attribute is a threshold timer for error recovery of the FC drivers. This Fibre Channel (FC) I/O attribute enhances the FC device driver control over the command timeout processing. The valid values of this attribute are as follows:

yes

Enables the I/O threshold attribute. When you enable this attribute, a threshold timer is started in the FC device driver that monitors the I/O operations in the pending queue of the FC device driver. If an I/O operation is pending for more than the **rw_timeout** seconds (the maximum read/write timeout specified by SCSI device driver), then the FC device driver completes that I/O operation, with the required status that is required by the SCSI device driver. The SCSI device driver controls the **rw_timeout** attribute for each I/O operation that the SCSI device driver is sending to the FC device driver.

no

Disables the I/O threshold attribute. This is the default value.

To enable error recovery for a FC SCSI device, enter the following command:

```
# chdev -U -l fscsi0 -a io_thrshld_tm=yes
```

where, the FC SCSI device instance is **fscsi0**.

To disable error recovery for a FC SCSI device, enter the following command:

```
# chdev -U -l fscsi0 -a io_thrshld_tm=no
```

where, the Fibre Channel SCSI device instance is **fscsi0**.

The **io_thrshld_tm** attribute supports concurrent update that means that you can update the attribute while FC SCSI device instance **fscsi0** is open and in use.

Path control module attributes

In addition to the AIX default path control modules (PCMs), a device-specific PCM might be supplied by a device vendor. The set of user changeable attributes is defined by the device vendor. A device-specific PCM can have device and path attributes.

The following are the device attributes for the AIX default PCMs:

algorithm

Determines the methodology by which the I/O is distributed across the paths for a device. The algorithm attribute has the following values:

Note: Some devices support only a subset of these values.

fail_over

Sends all I/O operations to a single path. If the path is marked as failed or disabled, the next available path is selected for sending all I/O operations. This algorithm maintains all the enabled paths in an ordered list based on the ascending values of the **path priority** attribute. The valid path that has the lowest path priority value is selected for each I/O operation.

round_robin

Distributes the I/O operations across multiple enabled paths. For devices that have active and passive paths, or preferred and non-preferred paths, only a subset of the paths are used for I/O operations. If a path is marked as failed or disabled, it is no longer used for sending I/O operations. The I/O operation is distributed based on the **path priority** attribute. Paths that have a higher path priority value receive a greater share of the I/O operations.

shortest_queue

Distributes the I/O operations across multiple enabled paths. For devices that have active and passive paths, or preferred and non-preferred paths, only a subset of the paths are used for I/O operations. This algorithm is similar to the **round_robin** algorithm. However, the **shortest_queue** algorithm distributes I/O operations based on the number of pending I/O operations on each path. The path that currently has the fewest pending I/O operations is selected for the next operation. The **path priority** attribute is ignored when the algorithm is set to **shortest_queue**.

hcheck_mode

Determines which paths must be checked when the health check capability is used. The attribute supports the following modes:

enabled

Sends the **healthcheck** command through paths that have a state of enabled. The mode does not send the **healthcheck** command through paths that have a state of disabled or missing.

failed

Sends the **healthcheck** command through paths that have a state of failed. The mode does not send the **healthcheck** command through paths that have a state of enabled, disabled, or missing.

nonactive

(Default) Sends the **healthcheck** command through paths that have no active I/O to the device, including paths that have a state of failed or enabled. The mode does not send the **healthcheck** command through paths that have a state of disabled or missing.

hcheck_interval

Defines how often the health check is performed on the paths for a device. The attribute supports a range 0 - 3600 seconds. When a value of 0 is selected, health checking is disabled.

Note: Health check is performed only if the disk is opened by some process and not yet closed. If no entity has the disk opened, the Path Control module does not check the paths even if the **hcheck_interval** attribute of that device is set to a nonzero value.

dist_tw_width

Defines the duration of a "time window". This is the time frame during which the distributed error detection algorithm cumulates I/Os returning with an error. The **dist_tw_width** attribute unit of measure is milliseconds. Lowering this attributes value decreases the time duration of each sample taken and decreases the algorithms sensitivity to small bursts of I/O errors. Increasing this attribute value increases the algorithms sensitivity to small bursts of errors and the probability of failing a path.

Note: >| You can modify the **dist_tw_width** attribute for an hdisk. The **dist_tw_width** is not displayed in the list of attributes of the **lsattr** command. |<

dist_err_percent

Defines the percentage of "time windows" having an error allowed on a path before the path is failed due to poor performance. The **dist_err_percent** has a range 0 - 100. The distributed error detection algorithm is disabled when the attribute is set to zero (0). The default setting is zero. The distributed error detection algorithm samples the fabric that connects the device to the adapter for errors. The algorithm calculates a percentage of samples with errors and will fail a path if the calculated value is larger than the **dist_err_percent** attribute value.

Note: >| You can modify the **dist_err_percent** attribute for an hdisk. The **dist_err_percent attribute** is not displayed in the list of attributes of the **lsattr** command. |<

The following is the path attribute for the AIX PCM:

path priority

Modifies the behavior of the algorithm methodology on the list of paths.

When the algorithm attribute value is **fail_over**, the paths are kept in a list. The sequence in this list determines which path is selected first and, if a path fails, which path is selected next. The sequence is determined by the value of the path priority attribute. A priority of 1 is the highest priority. Multiple paths can have the same priority value, but if all paths have the same value, selection is based on when each path was configured.

When the algorithm attribute value is **round_robin**, the **path priority** algorithm assigns a priority value to each path. The paths are selected for I/O operations in proportion to the path priorities. Therefore, paths with higher priority values are selected for more I/O operations. If all path priorities are the same, paths are selected equally.

timeout_policy

Adjusts the behavior of the PCM for the command timeouts and transport errors. When the **timeout_policy** is set to either **fail_path** or **disable_path**, the performance degradation might improve when an Multiple Path I/O (MPIO) device encounters intermittent storage area network (SAN) fabric issues on some paths to the device. The **timeout_policy** attribute has the following values:

retry_path

The first occurrence of a command timeout on the path does not cause immediate path failure. If a path that failed due to transport problems is recovered by a health check, the recovered path can be used immediately.

fail_path

The path fails on the first occurrence of a command timeout, assuming it is not the last path in the path group. If a path that failed due to transport problems recovers, the path is not used for read or write I/O operations until a period expires with no failures on that path. When this feature is enabled, a delay might occur before the read or write I/O is routed to paths that are recovered from a transport error.

fail_ctlr

This setting causes MPIO to switch from the preferred controller to the non-preferred controller more quickly. If you do not enable this setting, all of the paths to the preferred controller will fail before MPIO switches from the preferred controller to the non-preferred controller. When you enable this setting, MPIO switches when there are errors on two paths to the preferred controller. This setting is similar to the **fail_path** setting.

disable_path

The path fails on the first occurrence of a command timeout, assuming it is not the last path in the path group. If a path that failed due to transport problem recovers, the path is not used for read or write I/O until a period expires with no failures on that path. If this path continues to experience multiple command timeouts during a period, it might be disabled. Disabled paths remain disabled (and not usable) until you do one of the following actions: run the **chpath** command to enable the disabled path, reconfigure the affected disk, or reboot the system.

SAN replication attributes

The AIX Multiple Path I/O (MPIO) must be installed and the device must be using the AIX path control module (PCM). These attributes have dependencies on settings and features that are provided by the storage subsystem.

The following AIX attributes pertain to the behavior of the logical unit numbers (LUNs), which are replicated through the storage subsystems. All storage subsystems or microcode levels might not support these attributes. Clustering or high availability software, such as PowerHA SystemMirror is installed to coordinate management of the storage area network (SAN) replication across nodes in a cluster. The following attributes cannot be changed on a Virtual I/O Server (VIOS).

san_rep_cfg

Determines how a synchronous device that is using Peer-to-Peer Remote Copy (PPRC) is defined and configured in the AIX operating system. The **unique_id** value of the disk instance is affected by this attribute, and might change if the attribute value is altered. The **san_rep_cfg** attribute does not alter the state of the PPRC device on the storage subsystem. The attribute has the following values:

none [Default]

Configures LUNs in a synchronous device that is using PPRC as separate logical disk instances.

new

Defines and configures the synchronous device that is using PPRC as a single logical instance. The device is defined and configured only when no existing disk instances match any LUNs in the PPRC device.

new_and_existing

Defines and configures the synchronous device that is using PPRC as a single logical instance. If no logical disk instance represents the PPRC device, a new disk instance is defined.

migrate_disk

Defines and configures the synchronous device that is using PPRC as a single hdisk instance, and uses the selected logical disk instance for the device. The operation is only supported on devices that have the **san_rep_device** attribute set to either supported or detected. If the targeted device has the **san_rep_device** attribute set to supported, the operation works only if the SAN replication was set up on the storage since the last time the disk was configured. This operation is supported on disks that are opened and in use by the AIX operating systems if the device is not used as a repository disk within a cluster. The `unique_id` value for the affected disk is updated to reflect the ID of the PPRC device.

revert_disk

Configures an existing synchronous device that is using PPRC logical disk instance to a non-PPRC device hdisk instance. This operation is supported only on devices that have the **san_rep_device** attribute set to yes. The logical device name and special file of the targeted disk instance remains unchanged. The primary (source) LUN for a SAN replication device is used for the reverted hdisk instance. If the primary (source) LUN is not found or is unknown to the host, the secondary (destination) LUN is used for the reverted hdisk instance. This operation is supported on disks that are opened and in use by the AIX operating systems if the device is not used as a repository disk within a cluster. The `unique_id` value for the affected disk is updated to reflect the LUN ID.

The **none**, **new**, and **new_and_existing** values are meant to update the behavior for all devices of the same Object Data Manager (ODM) unique type. The **chdef** command is used to set the **none**, **new**, and **new_and_existing** values. The **chdef** command updates the default value for an attribute for all devices of the specified ODM unique type. The **chdef** command also updates the attribute value for the existing device instances that are already defined. For devices that are already configured when the **chdef** command is running, the change does not take effect until the system reboots or those devices are unconfigured and reconfigured. The **chdef** command requires the class, subclass, and type of the attribute. To determine the ODM unique type of the **san_rep_cfg** attribute, use the `lsattr -l hdisk# -F class,subclass,type` command. For example:

```
lsdev -l hdisk0 -F class,subclass,type
disk,fc, aixmpiods8k
chdef -a san_rep_cfg=none -c disk -s fc -t aixmpiods8k
chdef -a san_rep_cfg=new -c disk -s fc -t aixmpiods8k
chdef -a san_rep_cfg=new_and_existing -c disk -s fc -t aixmpiods8k
```

The **migrate_disk** and **revert_disk** values are meant to update behavior for a single and specified device instance. The **chdev** command must be used to set either the **migrate_disk** or **revert_disk** value for a specified device. The **chdev** command updates the value for only the specified device. For example:

```
chdev -a san_rep_cfg=migrate_disk -l hdisk0
chdev -a san_rep_cfg=revert_disk -l hdisk0
```

san_rep_device

Indicates that the logical disk instance is defined as a SAN replication device. This attribute is set during disk configuration, and might be stale if the state of the device is changed since the disk was configured. The attribute has the following values:

no

The device is not configured in the AIX operating system as a SAN replication device. This device does not meet the requirements to be configured as a SAN replication device.

supported

The device is not configured in the AIX operating system as a SAN replication device. This device meets the requirements to be configured as a SAN replication device. However, it is not currently set up as a SAN replication device on the storage subsystem.

detected

The device is not configured in the AIX operating system as a SAN replication device. The AIX operating system has detected that this device meets the requirements to be configured as a SAN replication device and is currently set up as a SAN replication device on the storage subsystem.

yes

The device is configured in the AIX operating system as a SAN replication device.

Communications adapter removal

Before you can remove or replace a hot-plug adapter, you must unconfigure that adapter.

Unconfiguring a communications adapter involves the following tasks:

- Closing all applications that are using the adapter you are removing or replacing
- Ensuring that all devices connected to the adapter are identified and stopped
- Listing all slots that are currently in use or a slot that is occupied by a specific adapter
- Identifying the adapter's slot location
- Displaying and removing interface information from the network interface list
- Making the adapter unavailable

To unconfigure communications adapters with the following procedures you must log in as **root**:

Unconfiguring Ethernet, Token-ring, FDDI, and ATM adapters

To unconfigure an Ethernet, Token-ring, FDDI, or ATM Adapter, perform the following steps:

1. Type `lsslot -c pci` to list all the hot-plug slots in the system unit and display their characteristics.
2. Type the appropriate SMIT command, shown in the following examples, to list installed adapters and show the current state of all the devices in the system unit:

Item	Description
smit lsdenet	To list Ethernet adapters
smit lsdtok	To list token-ring adapters
smit ls_atm	To list ATM adapters

The following naming convention is used for the different type of adapters:

Item	Description
Name	Adapter Type
atm0, atm1, ...	ATM adapter
ent0, ent1, ...	Ethernet adapter
tok0, tok1, ...	Token Ring adapter

3. Close all applications that are using the adapter you are unconfiguring.

To continue with this procedure, network dump locations must be disabled on the system. To look for and disable network dump locations, do the following:

- a) Type the following from a command line:

```
smit dump
```

- b) Select **Show Current Dump Devices**.

- c) Check whether any configured dump device shows a network location.

If not, exit SMIT and you are ready for step 4. To change a dump device to a local location, select **Cancel** or press F3 and continue with the following step.

- d) If the primary dump device shows a network location, change to a local location by selecting **Change the Primary Dump Device** and then enter the local location in the **Primary dump device** field.

- e) If the secondary dump device shows a network location, change to a local location by selecting **Change the Secondary Dump Device** and then enter the local location in the **Secondary dump device** field.
- f) When finished, click **OK** or press Enter.
4. Type `netstat -i` to display a list of all configured interfaces and determine whether your adapter is configured for TCP/IP. Output similar to the following displays:

```

Name  Mtu   Network  Address      IpKts  Ierrs  OpKts  Oerrs  Coll
lo0   16896 link#1    127.0.0.1    076   0      118    0      0
lo0   16896 127      127.0.0.1    076   0      118    0      0
lo0   16896 ::1      6.6.6.5      076   0      118    0      0
tr0   1492  link#2    8.0.5a.b8.b.ec 151   0      405    11     0
tr0   1492 19.13.97 19.13.97.106 151   0      405    11     0
at0   9180  link#3    0.4.ac.ad.e0.ad 0      0      0      0      0
at0   9180 6.6.6    6.6.6.5      0      0      0      0      0
en0   1500  link#5    0.11.0.66.11.1 212   0      1      0      0
en0   1500 8.8.8     8.8.8.106    212   0      1      0      0

```

Token-ring adapters can have only one interface. Ethernet adapters can have two interfaces. ATM adapters can have multiple interfaces.

5. Type the appropriate **ifconfig** command, shown in the following examples, to remove the interface from the network interface list.

Item	Description
<code>ifconfig en0 detach</code>	To remove the standard Ethernet interface
<code>ifconfig et0 detach</code>	To remove the IEEE 802.3 Ethernet interface
<code>ifconfig tr0 detach</code>	To remove a token-ring interface
<code>ifconfig at0 detach</code>	To remove an ATM interface

6. Type the appropriate **rmdev** command, shown in the following examples, to unconfigure the adapter and keep its device definition in the Customized Devices Object Class:

Item	Description
<code>rmdev -l ent0</code>	To unconfigure an Ethernet adapter
<code>rmdev -l tok1</code>	To unconfigure a token-ring adapter
<code>rmdev -l atm1</code>	To unconfigure an ATM adapter
<code>rmdev -p pci1</code>	To unconfigure the children of a PCI bus and all other devices under them while retaining their device definitions in the Customized Devices object class.

Note: To unconfigure the adapter and *remove* the device definition in the Customized Devices object class, you can use the **rmdev** command with the **-d** flag.



Attention: Do not use the **-d** flag with the **rmdev** command for a hot-plug operation unless your intent is to remove the adapter and not replace it.

Unconfiguring WAN adapters

You can unconfigure a WAN adapter.

To unconfigure a WAN Adapter:

1. Type `lsslot -c pci` to list all the hot-plug slots in the system unit and display their characteristics.
2. Type the appropriate SMIT command, shown in the following examples, to list installed adapters and show the current state of all the devices in the system unit:

Item	Description
smit 331121b9_ls	To list 2-Port Multiprotocol WAN adapters
smit riciophx_ls	To list ARTIC WAN adapters

The following naming convention is used for the different type of adapters:

Name	Adapter Type
dpmpa	2-Port Multiprotocol Adapter
riciop	ARTIC960 Adapter

3. Close all applications that are using the adapter you are unconfiguring.

To continue with this procedure, network dump locations must be disabled on the system. To look for and disable network dump locations, do the following:

- a) Type the following from a command line:

```
smit dump
```

- b) Select **Show Current Dump Devices**.
 - c) Check whether any configured dump device shows a network location.
If not, exit SMIT and you are ready for step 5 below. To change a dump device to a local location, select **Cancel** or press F3 and continue with the following step.
 - d) If the primary dump device shows a network location, change to a local location by selecting **Change the Primary Dump Device** and then enter the local location in the **Primary dump device** field.
 - e) If the secondary dump device shows a network location, change to a local location by selecting **Change the Secondary Dump Device** and then enter the local location in the **Secondary dump device** field.
 - f) When finished, click **OK** or press Enter.
4. Use the commands in the following table to unconfigure and remove the device drivers and emulator ports for these adapters:

Name	2-Port Multiprotocol adapter
smit rmhdlcdpmpdd	To unconfigure the device
smit rmsdlcscied	To unconfigure the SDLC COMIO emulator

Name	ARTIC960Hx PCI adapter
smit rmtsd	To unconfigure the device driver
smit rmtsdports	To remove an MPQP COMIO emulation port

Unconfiguring IBM 4-Port 10/100 Base-TX Ethernet PCI adapters

The 4-Port 10/100 Base-TX Ethernet PCI adapter has four ethernet ports and each port must be unconfigured before you can remove the adapter.

1. Type `lsslot -c pci` to list all the hot-plug slots in the system unit and display their characteristics.
2. Type `smit lsdnet` to list all the devices in the PCI subclass.

A message similar to the following displays:

```
ent1 Available 1N-00 IBM 4-Port 10/100 Base-TX Ethernet PCI Adapter (23100020) (Port 1)
ent2 Available 1N-08 IBM 4-Port 10/100 Base-TX Ethernet PCI Adapter (23100020) (Port 2)
ent3 Available 1N-10 IBM 4-Port 10/100 Base-TX Ethernet PCI Adapter (23100020) (Port 3)
ent4 Available 1N-18 IBM 4-Port 10/100 Base-TX Ethernet PCI Adapter (23100020) (Port 4)
```

3. Close all applications that are using the adapter you are unconfiguring.

To continue with this procedure, network dump locations must be disabled on the system. To look for and disable network dump locations, do the following:

- a) Type the following from a command line:

```
smit dump
```

- b) Select **Show Current Dump Devices**.
 - c) Check whether any configured dump device shows a network location.
If not, exit SMIT and you are ready for step 4. To change a dump device to a local location, select **Cancel** or press F3 and continue with the following step.
 - d) If the primary dump device shows a network location, change to a local location by selecting **Change the Primary Dump Device** and then enter the local location in the **Primary dump device** field.
 - e) If the secondary dump device shows a network location, change to a local location by selecting **Change the Secondary Dump Device** and then enter the local location in the **Secondary dump device** field.
 - f) When finished, click **OK** or press Enter.
4. Type `netstat -i` to display a list of all configured interfaces and determine whether your adapter is configured for TCP/IP.

Output similar to the following displays:

Name	Mtu	Network	Address	Ipkts	Ierrs	Opkts	Oerrs	Coll
lo0	16896	link#1		076	0	118	0	0
lo0	16896	127	127.0.0.1	076	0	118	0	0
lo0	16896	::1		076	0	118	0	0
tr0	1492	link#2	8.0.5a.b8.b.ec	151	0	405	11	0
tr0	1492	19.13.97	19.13.97.106	151	0	405	11	0
at0	9180	link#3	0.4.ac.ad.e0.ad	0	0	0	0	0
at0	9180	6.6.6	6.6.6.5	0	0	0	0	0
en0	1500	link#5	0.11.0.66.11.1	212	0	1	0	0
en0	1500	8.8.8	8.8.8.106	212	0	1	0	0

Ethernet adapters can have two interfaces, for example, `et0` and `en0`.

5. Use the **ifconfig** command to remove each interface from the network interface list.
For example, type `ifconfig en0 detach` to remove the standard Ethernet interface, and type `ifconfig et0` to remove the IEEE 802.3 interface.
6. Use the **rmdev** command to unconfigure the adapter and retain its device definition in the Customized Devices Object Class.
For example, `rmdev -l ent0`.

Note: To unconfigure the adapter and *remove* the device definition in the Customized Devices object class, you can use the **rmdev** command with the **-d** flag.



Attention: Do not use the **-d** flag with the **rmdev** command for a hot-plug operation unless your intent is to remove the adapter and not replace it.

Unconfiguring ATM adapters

You must unconfigure each LAN-emulated device before you can remove the adapter.

Classic IP and LAN emulation protocols can run over ATM adapters. LAN emulation protocol enables the implementation of emulated LANs over an ATM network. Emulated LANs can be Ethernet/IEEE 802.3, Token-ring/IEEE 802.5, and MPOA (MultiProtocol Over ATM).

To remove a LAN interface, do the following:

1. Type `lsslot -c pci` to list all the hot-plug slots in the system unit and display their characteristics.
2. Type `smit ls_atm` to list all the ATM adapters.

A message similar to the following displays:


```
.
.
atm0 Available 04-04 IBM PCI 155 Mbps ATM Adapter (14107c00)
atm1 Available 04-06 IBM PCI 155 Mbps ATM Adapter (14104e00)
```

3. Type `smit listall_atmle` to list all the LAN-emulated clients on the adapters.

A message similar to the following displays:

```
ent1 Available  ATM LAN Emulation Client (Ethernet)
ent2 Available  ATM LAN Emulation Client (Ethernet)
ent3 Available  ATM LAN Emulation Client (Ethernet)
tok1 Available  ATM LAN Emulation Client (Token Ring)
tok2 Available  ATM LAN Emulation Client (Token Ring)
```

All ATM adapters can have multiple emulated clients running on them.

4. Type `smit listall_mpoa` to list all the LAN-emulated clients on the adapters.

A message similar to the following displays:

```
mpc0 Available      ATM LAN Emulation MPOA Client
```

atm0 and atm1 are the physical ATM adapters. mpc0 is an MPOA-emulated client. ent1, ent2, ent3, tok1, and tok2 are LAN-emulated clients.

5. Type `entstat` to determine on which adapter the client is running.

A message similar to the following displays:

```
-----
ETHERNET STATISTICS (ent1) :
Device Type: ATM LAN EmulationATM Hardware Address: 00:04:ac:ad:e0:ad
.
.
.
ATM LAN Emulation Specific Statistics:
-----
Emulated LAN Name: ETHelan3
Local ATM Device Name: atm0
Local LAN MAC Address:
.
.
```

6. Close all applications that are using the adapter you are unconfiguring.

To continue with this procedure, network dump locations must be disabled on the system. To look for and disable network dump locations, do the following:

- a) Type the following from a command line:

```
smit dump
```

- b) Select **Show Current Dump Devices**.

- c) Check whether any configured dump device shows a network location.

If not, exit SMIT and you are ready for step 7. To change a dump device to a local location, select **Cancel** or press F3 and continue with the following step.

- d) If the primary dump device shows a network location, change to a local location by selecting **Change the Primary Dump Device** and then enter the local location in the **Primary dump device** field.

- e) If the secondary dump device shows a network location, change to a local location by selecting **Change the Secondary Dump Device** and then enter the local location in the **Secondary dump device** field.

- f) When finished, click **OK** or press Enter.

7. Use the `rmdev -l device` command to unconfigure the interfaces in the following order:

- Emulated interface = en1, et1, en2, et2, tr1, tr2 ...
- Emulated interface = ent1, ent2, tok1, tok2 ...

- Multiprotocol Over ATM (MPOA) = mpc0
 - ATM adapter = atm0
8. To unconfigure the SCSI adapter `scsi1` and all of its children while retaining their device definitions in the Customized Devices object class, type:

```
rmdev -R scsi1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
scsi1 Defined
```

9. To unconfigure just the children of the SCSI adapter `scsi1`, but not the adapter itself, while retaining their device definitions in the Customized Devices object class, type:

```
rmdev -p scsi1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
```

10. To unconfigure the children of PCI bus `pci1` and all other devices under them while retaining their device definitions in the Customized Devices object class, type:

```
rmdev -p pci1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
scsi1 Defined
ent0 Defined
```

Unconfiguring storage adapters

Before you can remove or replace a storage adapter, you must unconfigure that adapter.

To perform these tasks, you must log in as root user.

The following steps unconfigure SCSI and Fibre Channel storage adapters.

Unconfiguring a storage adapter involves the following tasks:

- Closing all applications that are using the adapter you are removing, replacing, or moving
- Unmounting file systems
- Ensuring that all devices connected to the adapter are identified and stopped
- Listing all slots that are currently in use or a slot that is occupied by a specific adapter
- Identifying the adapter's slot location
- Making parent and child devices unavailable
- Making the adapter unavailable

Unconfiguring SAS, SCSI, NVMe, and Fibre Channel adapters

Storage adapters are generally parent devices to media devices, such as disk or tape drives. Removing the parent requires that all attached child devices either be removed or placed in the define state.

To unconfigure SCSI and Fibre Channel adapters, perform the following steps:

1. Close all applications that are using the adapter you are unconfiguring.
2. Type `lsslot -c pci` to list all the hot plug slots in the system unit and display their characteristics.
3. Type `lsdev -C` to list the current state of all the devices in the system unit.

4. Type `umount` to unmount previously mounted file systems, directories, or files using this adapter. For additional information, see [Mount a JFS or JFS2](#).
5. Type `rmdev -l adapter -R` to make the adapter unavailable.



Attention: Do *not* use the **-d** flag with the **rmdev** command for hot plug operations because this will cause your configuration to be removed.

Unconfiguring async adapters

You can unconfigure an async adapter.

To perform these tasks, you must log in as root user.

The following are the steps for unconfiguring async adapters.

Before you can remove or replace an async adapter, you must unconfigure that adapter. Unconfiguring an async adapter involves the following tasks:

- Closing all applications that are using the adapter you are removing, replacing, or moving
- Ensuring that all devices connected to the adapter are identified and stopped
- Listing all slots that are currently in use or a slot that is occupied by a specific adapter
- Identifying the adapter's slot location
- Making parent and child devices unavailable
- Making the adapter unavailable

Procedure

Before you can replace or remove an async adapter, you must unconfigure the adapter and all the devices controlled by that adapter. To unconfigure the devices, you must terminate all the processes using those devices. Use the following steps:

1. Close all applications that are using the adapter you are unconfiguring.
2. Type `lsslot -c pci` to list all the hot plug slots in the system unit and display their characteristics.
3. Type `lsdev -C -c tty` to list all available tty devices and the current state of all the devices in the system unit.
4. Type `lsdev -C -c printer` to list all printer and plotter devices connected to the adapter.
5. Use the **rmdev** command to make the adapter unavailable.



Attention: Do *not* use the **-d** flag with the **rmdev** command for hot plug operations because this will cause your configuration to be removed.

Troubleshooting I/O devices

You can determine the cause of device problems.

Device software check

Correct a device software problem by:

- Checking the Error Log
- Listing All Devices
- Checking the State of a Device
- Checking the Attributes of a Device
- Changing the Attributes of a Device
- Using a Device with Another Application
- Defining a New Device

Error log check

Check the error log to see whether any errors are recorded for either the device, its adapter, or the application using the device. Go to [Error Logging Facility](#) for information about performing this check. Return to this step after completing the procedures.

Did you correct the problem with the device?

If you were not able to correct the correct the problem using the previous method, go to the next step (**Device listing**) to list all of the devices.

Device listing

Use the **lsdev -C** command to list all defined or available devices. This command shows the characteristics of all the devices in your system.

If the device is in the list of devices, go to the next step (**Device state check**) to check the state of the device.

If the device is not in the list of devices, define a new device (see **New device definition**, below).

Device state check

Find the device in the list generated from the **lsdev -C** command. Check whether the device is in the Available state.

If the device is in the Available state, go to the next step (**Device attribute check**) to check the device attributes.

If the device is not in the Available state, define a new device (see **New device definition**, below).

Device attribute check

Use the **lsattr -E -l DeviceName** command to list the attributes of your device.

The **lsattr** command shows attribute characteristics and possible values of attributes for devices in the system. Refer to the documentation for the specific device for the correct settings.

If the device attributes are set correctly, see **Device use with another application**, below.

If the device attributes are not set correctly, go to the next step, **Device attribute change**.

Device attribute change

Use the **chdev -l Name -a Attribute=Value** command to change device attributes.

The **chdev** command changes the characteristics of the device you specify with the **-l Name** flag.

If changing the attributes did not correct the problem with the device, go to the next step, **Device use with another application**.

Device use with another application

Try using the device with another application. If the device works correctly with another application, there might be a problem with the first application.

If the device worked correctly with another application, you might have a problem with the first application. Report the problem to your software service representative.

If the device did not work correctly with another application, go to the next step, **New device definition**.

New device definition

Note: You must either have root user authority or be a member of the security group to use the **mkdev** command.

Use the **mkdev** command to add a device to the system.

The **mkdev** command can either define and make available a new device or make available a device that is already defined. You can uniquely identify the predefined device by using any combination of the **-c**, **-s**, and **-t** flags.

If defining the device did not correct the problem, You can either stop and report the problem to your service representative or use a diagnostics program to test your device.

Checking the device connections

To check your device connections, perform the following steps:

1. Check that power is available at the electrical outlet.
2. Check that the device power cable is correctly attached to the device and to the electrical outlet.
3. Check that the device signal cable is attached correctly to the device and to the correct connection on the system unit.
4. For SCSI devices, check that the SCSI terminator is correctly attached and the SCSI address setting is correct.
5. For communications devices, check that the device is correctly attached to the communications line.
6. Check that the device is turned on.

Refer to the documentation for the specific device for cabling and configuring procedures and for further troubleshooting information.

Go to the next step if the steps in this topic have not corrected the problem.

Adapter removal problem resolution

You may receive error messages if the device is open when you use the **rmdev** command to unconfigure an adapter.

If the following type of message displays when you use the **rmdev** command to unconfigure an adapter, this indicates that the device is open, possibly because applications are still trying to access the adapter you are trying to remove or replace.

```
#rmdev -l ent0
Method error (/usr/lib/methods/ucfgent):
0514-062
Cannot perform the requested function because the
specified device is busy.
```

To resolve the problem, you must identify any applications that are still using the adapter and close them. These applications can include the following:

- TCP/IP
- SNA
- OSI
- IPX/SPX
- Novell NetWare
- Streams
- The generic data link control (GDLC)
 - IEEE Ethernet DLC
 - Token-ring DLC
 - FDDI DLC

Systems Network Architecture applications

Some SNA applications that may be using your adapter include:

- DB2®
- TXSeries® (CICS® & Encina)
- DirectTalk
- MQSeries®
- HCON
- ADSM

Streams applications

Some of the streams-based applications that may be using your adapter include:

- IPX/SPX
- Novell NetWare V4 and Novell NetWare Services 4.1
- Connections and NetBios for this operating system

Applications running on WAN adapters

Applications that may be using your WAN adapter include:

- SDLC
- Bisync
- ISDN

TCP/IP applications

All TCP/IP applications using the interface layer can be detached with the **ifconfig** command. This causes the applications using TCP/IP to time out and warn users that the interface is down. After you add or replace the adapter and run the **ifconfig** command to attach the interface, the applications resume.

Checking the ready state of a device

You can check to see if a device is in a ready state.

To determine whether the device is in a ready state, do the following:

1. Check that the device's Ready indicator is on.
2. Check that removable media, such as tape, diskette, and optical devices, are inserted correctly.
3. Check the ribbon, the paper supply, and the toner supply for printers and plotters.
4. Check that the write medium is write-enabled if you are trying to write to the device.

Did your checks correct the problem with the device? If your check of the device's ready state did not correct the problem, go to the next step.

Device diagnostics

To determine if a device is defective, run your hardware diagnostics.

If running hardware diagnostics fails to find a problem with your device, check the device software. If your device passes the diagnostic tests, you might have a problem with the way your device works with your system software. If it is possible that the preceding problem exists, report the problem to your software service organization.

Targeted device configuration

The **cfgmgr** command can be used with the **-c** flag as a connection option for a limited scope of targeted configuration of I/O devices.

Targeted configuration of FC and FCoE devices

The **cfgmgr -c** option is used with Fibre Channel (FC) and Fibre Channel over Ethernet (FCoE) adapters for targeted configuration.

The **cfgmgr** command can be used with the **-c** flag as a connection option for a limited scope of device configuration. For FC and FCoE adapters, the syntax is as follows:

```
cfgmgr -l fscsi0 -c "parameter=val[,parameter=val,...]"
```

By using the connection filter string, you can limit the scope of device discovery by using one or more of the following parameters:

Table 11. Parameters for the cfgmgr -c flag	
Parameter name	Description
ww_name	Target device world wide port name
node_name	Target device world wide node name
scsi_id	Target device N_Port ID that maps to the Small Computer System Interface (SCSI) ID for Fibre Channel Protocol (FCP) storage devices
lun_id	Logical unit number (LUN)

For example, the following command configures a single LUN of **lun_id** 0x1000000000000 at the storage target port that has the world wide port name 0x5001738000330191:

```
# cfgmgr -l fscsi0 -c "ww_name=0x5001738000330191,lun_id=0x1000000000000"
```

This scan occurs only for the **fscsi0** host adapter port.

Notes:

- The leading characters 0x in the parameter value are optional.
- All of the parameters must be represented as a hexadecimal number.

In the following example, only one parameter is specified:

```
# cfgmgr -l fscsi0 -c "lun_id=0x1000000000000"
```

This command scans all of the storage device ports on the storage area network (SAN) and configures this single logical unit for every SAN target port where this LUN exists.

Guidelines and rules for connection filter parameters

Consider the following points when you use the connection filter parameters:

- Targeted configuration for FC and FCoE devices applies to switch-attach environments only. If you specify a connection string that is directly attached to a target port, the connection fails with a message indicating that the child devices cannot be found.
- The **-c** flag is supported only when accompanied by the **-l** flag of the **cfgmgr** command that limits the scope of the command to a single **fscsiX** device at a time.
- If you specify **-?** as the connection string for the **-c** flag of the **cfgmgr** command, along with **-v** flag, the usage information is displayed.

- If you specify duplicate parameters (for example, **lun_id** listed twice), it results an error. No devices are detected.
- Any combination of **lun_id**, **scsi_id**, **ww_name**, and **node_name** parameters are allowed, except for duplicates. To uniquely identify a LUN, target, or storage node to be configured, you must specify one or preferably two parameters, although more are allowed. The following list specifies the parameter or combination of parameters that is required to uniquely identify a LUN, target, or storage node:
 - The **ww_name** and **lun_id** parameters uniquely identify a LUN on a target port to be configured.
 - The **scsi_id** and **lun_id** parameters uniquely identify a LUN on a target port to be configured.
 - The **node_name** and **lun_id** parameters configure a LUN for all target ports for a specific storage node. These parameters can configure the target ports only if all the target ports have the same **node_name** parameter, which might be true for some storage devices.
 - The **ww_name** parameter configures all LUNs for a specific target.
 - The **node_name** parameter configures all target ports for a specific storage node (only if all target ports have the same **node_name** parameter, which might be true for some storage devices).
 - The **lun_id** parameter configures a LUN on all target ports that are visible from that `fscsi` device.
- If more than two parameters are specified, the device configuration code uses this extra information to validate the device location. If any of the specified parameter values are in conflict with the reported values on the SAN, the command fails and no devices are configured.

Tape drives

The system management functions described here relate to tape drives.

Many of these functions alter or get information from the device configuration database, which contains information about the devices on your system. The device configuration database consists of the predefined configuration database, which contains information about all possible types of devices supported on the system, and the customized configuration database, which contains information about the particular devices currently on the system. For the operating system to make use of a tape drive, or any other device, the device must be defined in the customized configuration database and must have a device type defined in the predefined configuration database.

Tape drive attributes

You can adjust these tape drive attributes to meet the needs of your system.

The attributes can be displayed or changed using the SMIT, or commands (in particular, the **lsattr** and the **chdev** commands).

Each type of tape drive only uses a subset of all the attributes.

General information about each attribute

Block size

The block size attribute indicates the block size to use when reading or writing the tape. Data is written to tape in blocks of data, with inter-record gaps between blocks. Larger records are useful when writing to unformatted tape, because the number of inter-record gaps is reduced across the tape, allowing more data to be written. A value of **0** indicates variable length blocks. The allowable values and default values vary depending on the tape drive.

Device Buffers

Setting the Device Buffers attribute (using the **chdev** command) to `mode=yes` indicates an application is notified of write completion after the data has been transferred to the data buffer of the tape drive, but not necessarily after the data is actually written to the tape. If you specify the `mode=no`, an application is notified of write completion only after the data is actually written to the tape. Streaming mode cannot be maintained for reading or writing if this attribute is set to the `mode=no` value. The default value is `mode=yes`.

With the `mode=no` value, the tape drive is slower but has more complete data in the event of a power outage or system failure and allows better handling of end-of-media conditions.

Extended File Marks

Setting the Extended File Marks attribute (for the **chdev** command, the `extfm` attribute) to the `no` value writes a regular file mark to tape whenever a file mark is written. Setting this attribute to the `yes` value writes an extended file mark. For tape drives, this attribute can be set on. The default value is `no`. For example, extended filemarks on 8 mm tape drives use 2.2 MB of tape and can take up to 8.5 seconds to write. Regular file marks use 184 KB and take approximately 1.5 seconds to write.

To reduce errors when you use an 8 mm tape in append mode, use extended file marks for better positioning after reverse operations at file marks.

Retension

Setting the *Retensioning* attribute (for the **chdev** command, the `ret` attribute) to `ret=yes` instructs the tape drive to re-tension a tape automatically whenever a tape is inserted or the drive is reset. *Retensioning* a tape means to wind to the end of the tape and then rewind to the beginning of the tape to even the tension throughout the tape. *Retensioning* the tape can reduce errors, but this action can take several minutes. If you specify the `ret=no` value, the tape drive does not automatically re-tense the tape. The default value is `yes`.

Density Setting #1 and Density Setting #2

Density Setting #1 (for the **chdev** command, the `density_set_1` attribute) sets the density value that the tape drive writes when using special files `/dev/rmt*`, `/dev/rmt*.1`, `/dev/rmt*.2`, and `/dev/rmt*.3`. Density Setting #2 (for **chdev**, the `density_set_2` attribute) sets the density value that the tape drive writes when using special files `/dev/rmt*.4`, `/dev/rmt*.5`, `/dev/rmt*.6`, and `/dev/rmt*.7`. See [“Special files for tape drives” on page 210](#) for more information.

The density settings are represented as decimal numbers in the range **0** to **255**. A zero (**0**) setting selects the default density for the tape drive, which is usually the high density setting of the drive. Specific permitted values and their meanings vary with different types of tape drives. These attributes do not affect the ability of the tape drive to read tapes written in all densities supported by the tape drive. It is customary to set Density Setting #1 to the highest density possible on the tape drive and Density Setting #2 to the second highest density possible on the tape drive.

Reserve support

For tape drives that use the Reserve attribute (for the **chdev** command, the `res_support` attribute), specifying the value `res_support=yes` causes the tape drive to be reserved on the SCSI bus while it is open. If more than one SCSI adapter shares the tape device, this ensures access by a single adapter while the device is open. Some SCSI tape drives do not support the reserve or release commands. Some SCSI tape drives have a predefined value for this attribute so that reserve or release commands are always supported.

Variable Length Block Size

The Variable Length Block Size attribute (for the **chdev** command, the `var_block_size` attribute) specifies the block size required by the tape drive when writing variable length records. Some SCSI tape drives require that a nonzero block size be specified in their Mode Select data even when writing variable length records. The Block Size attribute is set to **0** to indicate variable length records. See the specific tape drive SCSI specification to determine whether or not this is required.

Data Compression

Setting the Data Compression attribute (for the **chdev** command, the `compress` attribute) to `compress=yes` causes the tape drive to be in compress mode, if the drive is capable of compressing data. If so, then the drive writes data to the tape in compressed format so that more data fits on a single tape. Setting this attribute to `no` forces the tape drive to write in native mode (non compressed). Read operations are not affected by the setting of this attribute. The default setting is `yes`.

Autoloader

Setting the Autoloader attribute (for the **chdev** command, the `autoload` attribute) to `autoload=yes` causes Autoloader to be active, if the drive is so equipped. If so, and another tape is available in the loader, any read or write operation that advances the tape to the end is automatically

continued on the next tape. Tape drive commands that are restricted to a single tape cartridge are unaffected. The default setting is yes.

Retry Delay

The Retry Delay attribute sets the number of seconds that the system waits after a command has failed before reissuing the command. The system may reissue a failed command up to four times. This attribute applies only to type OST tape drives. The default setting is 45.

Read/Write Timeout

The Read/Write Timeout or Maximum Delay for a READ/WRITE attribute sets the maximum number of seconds that the system allows for a read or write command to complete. This attribute applies only to type OST tape drives. The default setting is 144.

Return Error on Tape Change

The Return Error on Tape Change or Reset attribute, when set, causes an error to be returned on open when the tape drive has been reset or the tape has been changed. A previous operation to the tape drive must have taken place that left the tape positioned beyond beginning of tape upon closing. The error returned is a -1 and `errno` is set to EIO. Once presented to the application, the error condition is cleared. Also, re-configuring the tape drive itself will clear the error condition.

Attributes for 2.0 GB 4 mm tape drives (type 4mm2gb)

The following are the attributes for 2.0 GB 4 mm tape drives (type 4mm2gb).

Block size

The default value is 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Attributes with fixed values

If a tape drive is configured as a 2.0 GB 4 mm tape drive, the attributes for Retension, Reserve Support, Variable Length Block Size, Density Setting #1, and Density Setting #2 have predefined values that cannot be changed. The density settings are predefined because the tape drive always writes in 2.0 GB mode.

Attributes for 4.0 GB 4 mm tape drives (type 4mm4gb)

The following are the attributes for 4.0 GB 4 mm tape drives (type 4mm4gb).

Block size

The default value is 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Density setting #1 and Density setting #2

The user cannot change the density setting of this drive; the device reconfigures itself automatically depending on the Digital Data Storage (DDS) media type installed, as follows:

Media Type	Device Configuration
DDS	Read-only.
DDS	Read/write in 2.0 GB mode only.
DDS2	Read in either density; write in 4.0 GB mode only.
non-DDS	Not supported; cartridge will eject.

Data compression

The general information for this attribute applies to this tape drive type.

Attributes with fixed values

If a tape drive is configured as a 4.0 GB 4 mm tape drive, the Retension, Reserve Support, Variable Length Block Size, Density Setting #1, and Density Setting #2 attributes have predefined values that cannot be changed.

Attributes for 2.3 GB 8 mm tape drives (type 8mm)

The following are attributes for 2.3 GB 8 mm tape drives (type 8mm).

Block size

The default value is 1024. A smaller value reduces the amount of data stored on a tape.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

The general information for this attribute applies to this tape drive type.

Attributes with fixed values

If a tape drive is configured as a 2.3 GB 8mm tape drive, the Retension, Reserve Support, Variable Length Block Size, Data Compression, Density Setting #1, and Density Setting #2 attributes have predefined values which cannot be changed. The density settings are predefined because the tape drive always writes in 2.3 GB mode.

Attributes for 5.0GB 8mm tape drives (type 8mm5gb)

The following are the attributes for 5.0GB 8mm tape drives (type 8mm5gb).

Block size

The default value is 1024. If a tape is being written in 2.3 GB mode, a smaller value reduces the amount of data stored on a tape.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following settings apply:

Setting	Meaning
140	5 GB mode (compression capable)
21	5 GB mode noncompressed tape
20	2.3 GB mode
0	Default (5.0 GB mode)

The default values are 140 for Density Setting #1, and 20 for Density Setting #2. A value of 21 for Density Setting #1 or #2 permits the user to read or write a noncompressed tape in 5 GB mode.

Data compression

The general information for this attribute applies to this tape drive type.

Attributes with fixed values

If a tape drive is configured as a 5.0 GB 8 mm tape drive, the Retension, Reserve Support, and Variable Length Block Size attributes have predefined values which cannot be changed.

Attributes for 20000 MB 8 mm tape drives (self-configuring)

The following are attributes for 20000 MB 8 mm tape drives (self-configuring).

Block size

The default value is 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The drive can read and write data cartridges in 20.0 GB format. During a Read command, the drive automatically determines which format is written on tape. During a Write, the Density Setting determines which data format is written to tape.

The following settings apply:

Setting	Meaning
39	20 GB mode (compression capable)
0	Default (20.0 GB mode)

The default value is 39 for Density Setting #1 and Density Setting #2.

Data compression

The general information for this attribute applies to this tape drive type.

Attributes with fixed values

If a tape drive is configured as a 20.0 GB 8 mm tape drive, the Retension, Reserve Support, and Variable Length Block Size attributes have predefined values which cannot be changed.

Attributes for 35 GB tape drives (type 35gb)

The following are attributes for 35 GB tape drives (type 35gb).

Block size

The IBM 7205 Model 311 throughput is sensitive to block size. The minimum recommended block size for this drive is 32 KB. Any block size less than 32 KB restricts the data rate (backup or restore time). The following table lists recommended block sizes by command:

Supported Command	Default Block Size (Bytes)	RECOMMENDATION
BACKUP	32 K or 51.2 K (default)	Uses either 32 K or 51.2 K depending on whether Backup is by name or not. No change is required.
TAR	10 K	There is an error in the manual that states a 512 KB block size. Set the Blocking parameter to -N64 .
MKSYSB	See BACKUP	The MKSYSB command uses the BACKUP command. No change is required.
DD	n/a	Set the Blocking Parameter to bs=32K .
CPIO	n/a	Set the Blocking Parameter to -C64 .

Note: You must be aware of the capacity and throughput when you select a blocksize. Small blocksizes have a significant impact on performance and a minimal impact on capacity. The capacities of the 2.6 GB format (density) and 6.0 GB format (density) are impacted when you use smaller than the recommended blocksizes. For example, using a blocksize of 1024 bytes to back up 32 GB of data takes approximately 22 hours. Backing up the same 32 GB of data by using a blocksize of 32 KB takes approximately 2 hours.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following chart shows the Supported Data Cartridge type and Density Settings (in decimal and hex) for the IBM 7205-311 Tape Drive. When you perform a restore (read) operation, the tape drive automatically sets the density to match the written density. When you perform a backup (write) operation, you must set the density to match the Data Cartridge that you are using.

Supported Data Cartridges	Native Capacity	Compressed Data Capacity	SMIT Density Setting	HEX Density Setting
DLTtape III	2.6 GB	2.6 GB (No Compression)	23	17h
	6.0 GB	6.0 GB (No Compression)	24	18h
	10.0 GB	20.0 GB (Default for drive)	25	19h
DLTtapeIIIxt	15.0 GB	30.6 GB (Default for drive)	25	19h
DLTtapeIV	20.0 GB	40.0 GB	26	1Ah
	35.0 GB	70.0 GB (Default for drive)	27	1Bh

Note: If you request an unsupported native capacity for the data cartridge, the drive defaults to the highest supported capacity for the data cartridge that is loaded into the drive.

Data compression

The actual compression depends on the type of data that is being written (see previous table). A compression ratio of 2:1 is assumed for this compressed data capacity.

Attributes with fixed values

The general information for this attribute applies to this tape drive type.

Attributes for 150 MB 1/4-inch tape drives (type 150mb)

The following are attributes for 150 MB 1/4-inch tape drives (type 150mb).

Block size

The default block size is 512. The only other valid block size is 0 for variable length blocks.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

Writing to a 1/4-inch tape can only occur at the beginning of tape (BOT) or after blank tape is detected. If data exists on the tape, you cannot overwrite the data except at BOT. If you wish to add data to a tape that has been written and then rewound, you must space forward until the next file mark is detected, which causes the system to return an error. Only then can you start writing again.

Retension

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following settings apply:

Setting	Meaning
16	QIC-150

Setting	Meaning
15	QIC-120
0	Default (QIC-150), or whatever was the last density setting by a using system.

The default values are 16 for Density Setting #1, and 15 for Density Setting #2.

Attributes with fixed values

If a tape drive is configured as a 150 MB 1/4-inch tape drive, attributes for Extended File Marks, Reserve Support, Variable Length Block Size, and Data Compression have predefined values which cannot be changed.

Attributes for 525 MB 1/4-inch tape drives (type 525mb)

The following are attributes for 525 MB 1/4-inch tape drives (type 525mb).

Block size

The default block size is 512. The other valid block sizes are 0 for variable length blocks, and 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

Writing to a 1/4-inch tape can only occur at the beginning of tape (BOT) or after blank tape is detected. If data exists on the tape, you cannot overwrite the data except at BOT. If you want to add data to a tape that has been written and then rewound, you must space forward until the next file mark is detected, which causes the system to return an error. Only then can you start writing again.

Retension

Writing to a 1/4-inch tape can only occur at the beginning of tape (BOT) or after blank tape is detected. If data exists on the tape, you cannot overwrite the data except at BOT. If you want to add data to a tape that has been written and then rewound, you must space forward until the next file mark is detected, which causes the system to return an error. Only then can you start writing again.

Density setting #1 and density setting #2

The following settings apply:

Setting	Meaning
17	QIC-525*
16	QIC-150
15	QIC-120
0	Default (QIC-525), or whatever was the last density setting by a using system.

* QIC-525 is the only mode that supports the 1024 block size.

The default values are 17 for Density Setting #1, and 16 for Density Setting #2.

Attributes with fixed values

If a tape drive is configured as a 525 MB 1/4-inch tape drive, the Extended File Marks, Reserve Support, Variable Length Block Size, and Data Compression attributes have predefined values which cannot be changed.

Attributes for 1200 MB 1/4-inch tape drives (type 1200mb-c)

The following are attributes for 1200 MB 1/4-inch tape drives (type 1200mb-c).

Block size

The default block size is 512. The other valid block sizes are 0 for variable length blocks, and 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

Writing to a 1/4-inch tape can only occur at the beginning of tape (BOT) or after blank tape is detected. If data exists on the tape, you cannot overwrite the data except at BOT. If you wish to add data to a tape that has been written and then rewound, you must space forward until the next file mark is detected, which causes the system to return an error. Only then can you start writing again.

Retention

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following settings apply:

Setting	Meaning
21	QIC-1000*
17	QIC-525*
16	QIC-150
15	QIC-120
0	Default (QIC-1000), or whatever was the last density setting by a using system.

* QIC-525 and QIC-1000 are the only modes that support the 1024 block size.

The default values are 21 for Density Setting #1, and 17 for Density Setting #2.

Attributes with fixed values

If a tape drive is configured as a 1200 MB 1/4-inch tape drive, the Extended File Marks, Reserve Support, Variable Length Block Size, and Data Compression attributes have predefined values which cannot be changed.

Attributes for 12000 MB 4 mm tape drives (self-configuring)

The following are attributes for 12000 MB 4 mm tape drives (self-configuring).

Block size

The IBM 12000 MB 4 mm tape drive's throughput is sensitive to block size. The minimum recommended block size for this drive is 32 KB. Any block size less than 32 KB restricts the data rate (backup or restore time). The following table lists recommended block sizes by command:

Supported Command	Default Block Size (Bytes)	Recommendation
BACKUP	32 K or 51.2 K (default)	Uses either 32 K or 51.2 K depending on whether Backup is by name or not. No change is required.
TAR	10 K	There is an error in the manual that states a 512 KB block size. Set the Blocking parameter to -N64 .
MKSYSB	See BACKUP	The MKSYSB command uses the BACKUP command. No change is required.
DD		Set the Blocking parameter to bs=32K.
CPIO		Set the Blocking parameter to -C64.

Note: You must be aware of the capacity and throughput when you select a block size. Small block sizes have a significant impact on performance and a minimal impact on capacity.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following chart shows the supported data cartridge type and density settings (in decimal and hex) for the IBM 12000 MB 4 mm tape drive. When you perform a restore (read) operation, the tape drive automatically sets the density to match the written density. When you perform a backup operation (write), you must set the density to match the data cartridge you are using.

Supported Data Cartridges	Native Capacity	Compressed Data Capacity	SMIT Density Setting	HEX Density Setting
DDS III	2.0 GB	4.0 GB	19	13h
DDS2	4.0 GB	8.0 GB	36	24h
DDS3	12.0 GB	24.0 GB	37	25h

Note: If you request an unsupported native capacity for the data cartridge, the drive defaults to the highest supported capacity for the data cartridge that is loaded into the drive.

Data compression

The actual compression depends on the type of data that is being written (see previous table). A compression ratio of 2:1 is assumed for this compressed data capacity.

Attributes with fixed values

The general information for this attribute applies to this tape drive type.

Attributes for 13000 MB 1/4-inch tape drives (self-configuring)

The following are the attributes for 13000 MB 1/4-inch tape drives (self-configuring).

Block size

The default block size is 512. The other valid block sizes are 0 for variable length blocks, and 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

Writing to a 1/4-inch tape can only occur at the beginning of tape (BOT) or after blank tape is detected. If data exists on the tape, you cannot overwrite the data except at BOT. If you wish to add data to a tape that has been written and then rewound, you must space forward until the next file mark is detected, which causes the system to return an error. Only then can you start writing again.

Retention

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following settings apply:

Setting	Meaning
33	QIC-5010-DC*
34	QIC-2GB*
21	QIC-1000*
17	QIC-525*
16	QIC-150
15	QIC-120
0	Default (QIC-5010-DC)*

* QIC-525, QIC-1000, QIC-5010-DC, and QIC-2GB are the only modes that support the 1024 block size.

The default values are 33 for Density Setting #1, and 34 for Density Setting #2.

Attributes with fixed values

If a tape drive is configured as a 13000 MB 1/4-inch tape drive, the Extended File Marks, Reserve Support, and Variable Length Block Size attributes have predefined values which cannot be changed.

Attributes for 1/2-inch 9-track tape drives (type 9trk)

The following are the attributes for 1/2-inch 9-track tape drives (type 9trk).

Block size

The default block size is 1024.

Device buffers

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The following settings apply:

Setting	Meaning
3	6250 bits per inch (bpi)
2	1600 bpi
0	Whichever writing density was used previously

The default values are 3 for Density Setting #1, and 2 for Density Setting #2.

Attributes with fixed values

If a tape drive is configured as a 1/2-inch 9-track tape drive, the Extended File Marks, Retension, Reserve Support, Variable Length Block Size, and Data Compression attributes have predefined values that cannot be changed.

Attributes for 3490e 1/2-inch cartridge (type 3490e)

The following are attributes for the 3490e 1/2-inch cartridge (type 3490e).

Block size

The default block size is 1024. This drive features a high data transfer rate, and block size can be critical to efficient operation. Larger block sizes can greatly improve operational speeds, and in general, the largest possible block size should be used.

Note: Increasing the block value can cause incompatibilities with other programs on your system. If this occurs, you receive the following error message while running those programs:

```
A system call received a parameter that is not valid.
```

Device buffers

The general information for this attribute applies to this tape drive type.

Compression

The general information for this attribute applies to this tape drive type.

Autoloader

This drive features a tape sequencer, an autoloader that sequentially loads and ejects a series of tape cartridges from the cartridge loader. For this function to operate correctly, the front panel switch should be in the AUTO position and the Autoloader attribute must be set to yes.

Attributes for other SCSI tapes (type ost)

The following are attributes for other SCSI tapes (type ost).

Block size

The system default is 512, but this should be adjusted to the default block size for your tape drive. Typical values are 512 and 1024. 8 mm and 4 mm tape drives usually use 1024 and waste space on the tape if the block size attribute is left at 512. A value of 0 indicates variable block size on some drives.

Device buffers

The general information for this attribute applies to this tape drive type.

Extended file marks

The general information for this attribute applies to this tape drive type.

Density setting #1 and density setting #2

The default value is 0 for both of these settings. Other values and their meanings vary for different tape drives.

Reserve support

The default value is no. This can be set to yes, if the drive supports reserve/release commands. If you are unsure, no is a safer value.

Variable length block size

The default variable length block size value is 0. Nonzero values are used primarily on quarter inch cartridge (QIC) drives. See the SCSI specification for the particular tape drive for advice.

Retry delay

This attribute applies exclusively to type ost tape drives.

Read/write timeout

This attribute applies exclusively to type ost tape drives.

Attributes with fixed values

If a tape drive is configured as an Other SCSI tape drive, the attributes for Extended File Marks, Retension, and Data Compression have predefined values which cannot be changed.

MPIO tape attributes

MPIO-supported tape devices will have additional attributes listed under MPIO device attributes.

Special files for tape drives

There are several special files associated with each tape drive known to the operating system.

Writing to and reading from files on tapes is done by using rmt special files. These special files are /dev/rmt*, /dev/rmt*.1, /dev/rmt*.2, through /dev/rmt*.7. The rmt* is the logical name of a tape drive, such as rmt0, rmt1, and so on.

By selecting one of the special files associated with a tape drive, you make choices about how the I/O operations related to the tape drive will be performed.

Item	Description
Density	You can select whether to write with the tape drive Density Setting #1 or with the tape drive Density Setting #2. The values for these density settings are part of the attributes of the tape drive. Because it is customary to set Density Setting #1 to the highest possible density for the tape drive and Density Setting #2 to the next highest possible density for the tape drive, special files that use Density Setting #1 are sometimes referred to as high density and special files that use Density Setting #2 sometimes are referred to as low density, but this view is not always correct. When reading from a tape, the density setting is ignored.

Item	Description
Rewind-on-Close	You can select whether the tape is rewound when the special file referring to the tape drive is closed. If rewind-on-close is selected, the tape is positioned at the beginning of the tape when the file is closed.
Retension-on-Open	You can select whether the tape is retensioned when the file is opened. Retensioning means winding to the end of the tape and then rewinding to the beginning of the tape to reduce errors. If retension-on-open is selected, the tape is positioned at the beginning of the tape as part of the open process.

The following table shows the names of the `rmt` special files and their characteristics.

Special File	Rewind on Close	Retension on Open	Density Setting
<code>/dev/rmt*</code>	Yes	No	#1
<code>/dev/rmt*.1</code>	No	No	#1
<code>/dev/rmt*.2</code>	Yes	Yes	#1
<code>/dev/rmt*.3</code>	No	Yes	#1
<code>/dev/rmt*.4</code>	Yes	No	#2
<code>/dev/rmt*.5</code>	No	No	#2
<code>/dev/rmt*.6</code>	Yes	Yes	#2
<code>/dev/rmt*.7</code>	No	Yes	#2

Suppose you want to write three files on the tape in tape drive `rmt2`. The first file is to be at the beginning of the tape, the second file after the first file, and the third file after the second file. Further, suppose you want Density Setting #1 for the tape drive. The following list of special files, in the order given, could be used for writing the tape.

1. `/dev/rmt2.3`
2. `/dev/rmt2.1`
3. `/dev/rmt2`

These particular special files are chosen because:

- `/dev/rmt2.3` is chosen as the first file because this file has Retension-on-Open, which ensures that the first file is at the beginning of the tape. Rewind-on-Close is not chosen because the next I/O operation is to begin where this file ends. If the tape is already at the beginning when the first file is opened, using the `/dev/rmt2.1` file as the first file would be faster since time for retensioning the tape is eliminated.
- `/dev/rmt2.1` is chosen for the second file because this file has neither Retension-on-Open nor Rewind-on-Close chosen. There is no reason to go to the beginning of the tape either when the file is opened or when it is closed.
- `/dev/rmt2` is chosen for the third and final file because Retension-on-Open is not wanted since the third file is to follow the second file. Rewind-on-Close is selected because there are no plans to do any more writing after the third file on the tape. The next use of the tape will begin at the beginning of the tape.

Besides controlling tape operations by choosing a particular `rmt` special file, you can use the `tctl` command to control tape operations.

USB device support

The AIX operating system supports Universal Serial Bus (USB) devices.

The AIX operating system supports the following types of third-party USB devices and IBM USB devices:

- USB mass storage class
 - Flash drive
 - Disk drive
 - Optical drive (Blu-ray, DVD, and CD-ROM)
 - Removable Disk Drives ([RDX](#))
 - External and portable hard disk drives
- USB Human Interface Device Class (HID)
 - Keyboard
 - Mouse

Note: The AIX operating system does not support USB printers.

> Starting from AIX 7.3 Technology Level 1, USB mass storage devices that have a capacity greater than 2 TB are supported, but have the following limitations:

- The accessible capacity for USB devices (block size 512 bytes) that have greater than 2 TB size is limited to the first 2 TB. The accessible capacity for USB devices (block size 4 KB) that have greater than 16 TB size is limited to the first 16 TB. You can get the block size of the USB device by using the **getconf** command.
- USB devices that have capacity greater than 2 TB do not support the **mksysb** and **udfcreate** commands regardless of the block size of the USB device.

In AIX 7.3 Technology Level 1, and later, Seagate and Western Digital USB devices were used to validate the support of USB devices that have capacity greater than 2 TB. <

> The AIX operating system does not support all the possible third-party USB devices because some third-party USB devices might not be recognized by the AIX operating system. Consider the following scenarios when the AIX operating system does not support the third-party USB devices:

- Insufficient power supply from the Power® Systems USB port.
- Missing quirks for the USB device. Third-party USB devices might have to add quirks if USB devices are not configured or are not working by default. For more information about adding quirks, see [AIX USB device Quirks](#).

<

USB flash drive support

Beginning with AIX 5.3 with the Technology Level 5300-09 and AIX 6.1 with the Technology Level 6100-02, Universal Serial Bus (USB) flash drives are supported.

Support for these devices is included in the following device package:

```
devices.usbif.08025002
```

AIX support for USB flash drives is validated against a sample of industry standard OEM USB flash drives. Device drivers for the AIX USB-host controller support USB 2.0. USB flash drives are configured with logical names, such as `usbms0` and `usbms1`, and they present both raw and block special files. For example, the raw special file for `usbms0` is `/dev/rusbms0`, and the block special file is `/dev/usbms0`. Before AIX Version 5.3 with the 5300-11 Technology Level and AIX Version 6.1 with the 6100-04 Technology Level, USB flash drives were configured as `/dev/flashdrive0`.

The International Organization for Standardization (ISO) file system (read-only ISO 9660) is supported on these drives. You can create a system backup on the drives by using the **tar** command, **cpio** command, or the backup or restore archives. You can also use the **dd** command to add the ISO images to the drives.

The AIX operating system does not support plug-and-play for USB flash drives. To make a flash drive available to AIX users, a root user must connect the drive to a system USB port and run the following command:

```
cfgmgr -l usb0
```



Attention: Use caution when you remove the flash drives from ports. If the drives are not properly closed or unmounted before you remove them, data on the drives can be corrupted.

After you remove the drives, they remain in the available state in the Object Data Manager (ODM) until the root user runs the following command:

```
rmdev -l usbmsn
```

When a drive is in the available state, you can reconnect it to the system, and it can be remounted or reopened. If a drive is disconnected from a system USB port while it is still open to a user, that drive is not reusable until the user closes and reopens it.

USB Blu-ray drive read-only support

AIX Version 6.1 with the 6100-06 Technology Level, and later, recognizes and configures USB attached Blu-ray drives.

This feature is included in the following device package:

```
devices.usbif.08025002
```

The capability of the AIX operating system to read Blu-ray media is validated against a sample of industry standard original equipment manufacturer (OEM) USB Blu-ray drives.

USB Blu-ray drives are configured by using logical names, such as `cd0` and `cd1`. The drives present both raw and block special files. For example, the raw special file for `cd0` is `/dev/rxcd0` and the block special file is `/dev/cd0`.

The read-only capability is provided for the International Organization for Standardization (ISO) file system (read-only ISO 9660), the Universal Disk Format (UDF) file system (Version 2.01, or earlier), and standard optical media access commands, such as **dd** and **tar**.

The AIX operating system does not support the write operation to CD, DVD, or Blu-ray media present in the USB Blu-ray drive. Although the write operation is not prevented (if the drive is write-capable), IBM does not provide support for any issues that are encountered during the write operation.

The AIX operating system does not support plug-and-play for USB Blu-ray drives. To make a USB Blu-ray drive available to AIX users, a root user must connect the drive to the USB port of the system and run the following command:

```
cfgmgr -l usb0
```

After the drive is removed, the drive remains in the available state in the Object Data Manager (ODM) database until the root user runs the following command:

```
rmdev -l cdn
```

When a drive is in the available state, you can reconnect it to the system. If a drive is disconnected from a system USB port while it is still open to a user, you cannot use that drive until you close and reopen it.

> AIX USB device quirks

Starting from AIX 7.3 Technology Level 1, the AIX operating system provides *quirks* to support various third-party USB mass storage devices.

The following quirks are supported starting from AIX 7.3 for USB mass storage devices:

delay_doorbell

If this quirk is set, the ringing of the adapter doorbell is delayed by 1 ms.

cbw_csw_order

If this quirk is set, the data packets are sent only in the order of command block wrapper (CBW), data, and command status wrapper (CSW). Most of the third-party devices, for example, Seagate and Western Digital, need this quirk to operate correctly with the AIX USB stack.

Note: The AIX base operating system provides predefined quirks for all Seagate and Western Digital devices. Therefore, you do not need to add quirks for these devices explicitly. For USB devices from other vendors, a quirk must be added if USB devices are not configured by default.

Creating a quirk entry for a specific third-party USB device

To enable the **cbw_csw_order** quirk for a vendor with vendor ID 0xVVVV and a product with product ID 0xPPPP, use the following example quirk entry:

```
PdAt:
  uniquetype = "usbms/usbif/0806500b"
  attribute = "quirk"
  deflt = "VVVV_PPPP"
  values = "cbw_csw_order"
  width = ""
  type = "R"
  generic = ""
  rep = "sl"
  nls_index = 0
```

Creating a quirk entry for all the USB devices from a specific vendor

To enable the **cbw_csw_order** quirk for all the USB devices from a vendor with vendor ID 0xVVVV, use the following example quirk entry:

```
PdAt:
  uniquetype = "usbms/usbif/0806500b"
  attribute = "quirk"
  deflt = "VVVV_all "
  values = "cbw_csw_order "
  width = ""
  type = "R"
  generic = ""
  rep = "sl"
  nls_index = 0
```

Adding a quirk for a USB device

If a third-party USB device is not configured on an AIX operating system, a quirk can be added according to the format shown in the preceding examples. The objects for the quirk entry for a specific USB device or a set of USB devices must be added in a PdAt object class. You must add a quirk entry for a USB device by using the **odmadd** command.

To add a quirk for a USB device that is not recognized by an AIX operating system or is not getting configured on an AIX operating system, run the following command:

1. Unconfigure the USB devices by running the following command:

```
rmdev -Rl usb0
```

2. Create a file for the quirk entry for a new USB device.

3. Add the quirk entry for the USB device by running the following command:

```
odmadd <file name>
```

4. Plug in the USB device if it is not plugged in already.

5. Configure the USB device by running the following command:

```
cfgmgr -vl usb0
```

⏪

Caching storage data

The server-side caching of storage data is supported on the cache devices.

The cache devices can be one of the following types of devices:

- Server-attached flash devices, such as built-in solid-state drive (SSD) in the server.
- Flash devices that are directly attached to the server by using Serial Attached SCSI (SAS) controllers.
- Flash resources in the storage area network (SAN).
- > Virtual Persistent Memory (vPMEM) disk devices. ⏪

The AIX operating system must identify the device as a flash device for it to be used as a cache device. The AIX operating system determines if a device is a flash device by using the **MEDIUM ROTATION RATE** field in the SCSI Block Device Characteristics Vital Product Data (VPD) page. If a device does not support that page or puts a value other than 0001h Non-rotating medium in the **MEDIUM ROTATION RATE** field, then the device cannot be used as a cache device.

Storage data caching concept

You can cache the storage data dynamically (start or stop caching) while the workload is running. The workload need not be brought down to an inactive state to perform the caching operation.

The following terms are used to explain the caching concept:

Cache device

A cache device is a solid-state drive (SSD), a flash disk, or a Virtual Persistent Memory (vPMEM) disk that is used for caching.

Cache pool

A cache pool is a group of cache devices that is used only for storage caching.

Cache partition

A cache partition is a logical cache device that is created from the cache pool.

Target device

A target device is a storage device that is being cached.

A single cache partition can be used to cache one or more target devices. When a target device is cached, all the read requests for the device blocks are routed to the caching software. If a specific block is found in the cache, the I/O request is processed from the cache device. If the requested block is not found in the cache, or if it is a write request, the I/O request returns to the target device.

Advantages of storage data caching

Server-side caching of storage data can increase virtualization density, especially when the storage subsystem is congested.

Storage data caching has the following advantages:

Latency

Analytical and transactional workloads have reduced query-response time because the solid-state drive (SSD) storage or Virtual Persistent Memory (vPMEM) has lower latencies. If you use server-side caching, the average latency for a transactional workload can be reduced by half.

Throughput

Online transaction processing (OLTP) workloads have higher transaction rates because the SSD storage or vPMEM provides better throughput.

Write throughput

In environments where the storage area network (SAN) is congested, the flash device or vPMEM, which is used as a cache, can offload a significant percentage of read requests. When read requests are offloaded, the SAN can have better write throughput, and can effectively serve a larger number of clients and hosts.

Smaller memory footprint

If a flash cache device is configured, some workloads can perform even with a lower memory footprint.

Limitations for storage data caching

Ensure that you understand the limitations and additional configuration requirements to use the caching feature. You must also consider the application restrictions for the target devices that must be cached.

Consider the following limitations for caching the storage data:

- The caching software is configured as a read-only cache, which means that only read requests are processed from the flash solid-state drive (SSD) or Virtual Persistent Memory (vPMEM). All write requests are processed by the original storage device.
- Data that is written to the storage device is not populated in the cache automatically. If the write operation is performed on a block that is in the cache, the existing data in the cache is marked as invalid. The same block reappears in the cache based on the frequency and recency of the access to the block.
- The caching software loads data into the cache based on local read patterns, and invalidates the cache entries locally. The target devices must not be shared by more than one LPAR concurrently. The target devices cannot be part of any clustered storage such as Oracle Real Application Clusters (RAC), DB2 pureScale®, and General Parallel File System (GPFS). Target devices that are part of a high-availability cluster can be cached only if the access specifies that a single host is reading or writing data from the target device at a time and caching is enabled only on the active node.
- The cache disk can be provisioned either to an AIX LPAR or to a Virtual I/O Server (VIOS) LPAR. Cache devices cannot be shared.
- The caching software must open the target devices to intercept any I/O requests to the target devices. If a workload needs to open the target device exclusively after caching is started, the exclusive open operation fails. In these instances, caching must be stopped and restarted after the workload starts. An example for exclusive open operation is setting the physical volume identifier (PVID) for target disks.
- If the disks are used as target devices, the **reserve_policy** attribute of the disk must not be set to `single_path`.
- When the caching operation is started for a target device, the cache engine logic delays the promotion of data into the cache. This delay is required to ensure that all outstanding I/O operations on the target device, which are issued before the caching operation is started, are completed before starting the caching operation. The exact time of delay is calculated internally based on the number of available paths and the **rw_timeout** attribute (if any) of the target disk. If the internally calculated time must be overridden by a user-defined time, you can set the `DEFAULT_IO_DRAIN_TIMEOUT_PD` environment variable in the `/etc/environment` file to a custom timeout value, in seconds.
- You cannot run the **cache_mgt** command to perform caching on target devices when the encryption is enabled for any physical volumes.
- NVMe devices cannot be used as target disks.

- >|vPMEM disk devices cannot be used as target disks.|<
- Additional memory is required on each AIX logical partition (LPAR) because the caching software manages metadata on each cache block. Reducing the number of storage protect keys might also be required to ensure that the caching software can allocate a sufficient contiguous block of memory.

Note: >|Starting with AIX 7.3, Technology Level 3, you can start the cache irrespective of the storage keys. However, the memory requirements are still applicable.|<

Table 12. Minimum required memory and maximum allowed storage protect keys to be able to start a flash cache partition of the indicated size		
Cache partition size	Storage protect keys	Required memory
<=20 TB	17 - 31	4 GB - ~15 GB
<=40 TB	9 - 16	~30 GB
<=90 TB	1 - 8	~60 GB
>90 TB	0	>60 GB

Components of storage data caching

The caching software consists of cache management and cache engine components.

Cache management

You can manage the storage data caching by using the **cache_mgt** command, which is available on the AIX operating system and on the Virtual I/O Server (VIOS). You can use the **cache_mgt** command to perform the following tasks:

- To create and partition a cache pool.
- To assign the cache partition to a target device or the AIX logical partition (LPAR) as a virtual Small Computer System Interface (vSCSI) device.
- To start and stop the caching operation.

Cache engine

The cache engine is the most essential part of the caching software. The cache engine decides which blocks in the storage must be cached, and whether the data must be retrieved from the cache or the primary storage.

The caching algorithm is based on a populate-on-read mechanism that fills the cache with data that has spatial locality (near other blocks that are read recently). The caching algorithm fills data in the cache more quickly when the cache is empty.

All the blocks in the cache are monitored to check how often they are read, and a heat map is generated. The heat map considers both frequency and recentness of access. When the cache is fully populated, new entries are added to the cache only if the new block is warmer than the coldest block in the cache. The coldest block is removed from the cache, and the new entry is added.

The aggressive population ensures short warm-up times that make the cache effective as soon as it is enabled. The removal policy, which is based on the heat map, ensures that the caching is dynamic and adjusts to changing workload patterns.

Configuring storage data caching

In the AIX operating system, server-side caching of flash devices is supported in several distinct configurations. These configurations differ on how the cache device is provisioned to the AIX logical partition (LPAR).

The server-side caching supports the following modes in the AIX operating system:

- Dedicated mode
- Virtual mode
- N_Port ID Virtualization (NPIV) mode

Caching storage data in dedicated mode

In the dedicated mode, the cache device is directly provisioned to the AIX logical partition (LPAR).

You must create a cache pool, and then a cache partition on the cache device. Only one cache partition can be created in a dedicated cache device. You can use the cache partition to cache any number of target devices on the AIX LPAR. The LPAR is not mobile because the cache device is dedicated to this LPAR. If the LPAR needs to be migrated to another server, you must manually stop the caching and unconfigure the cache device before the migration.

The following figure shows an example of caching configuration on an AIX LPAR for a dedicated cache device.

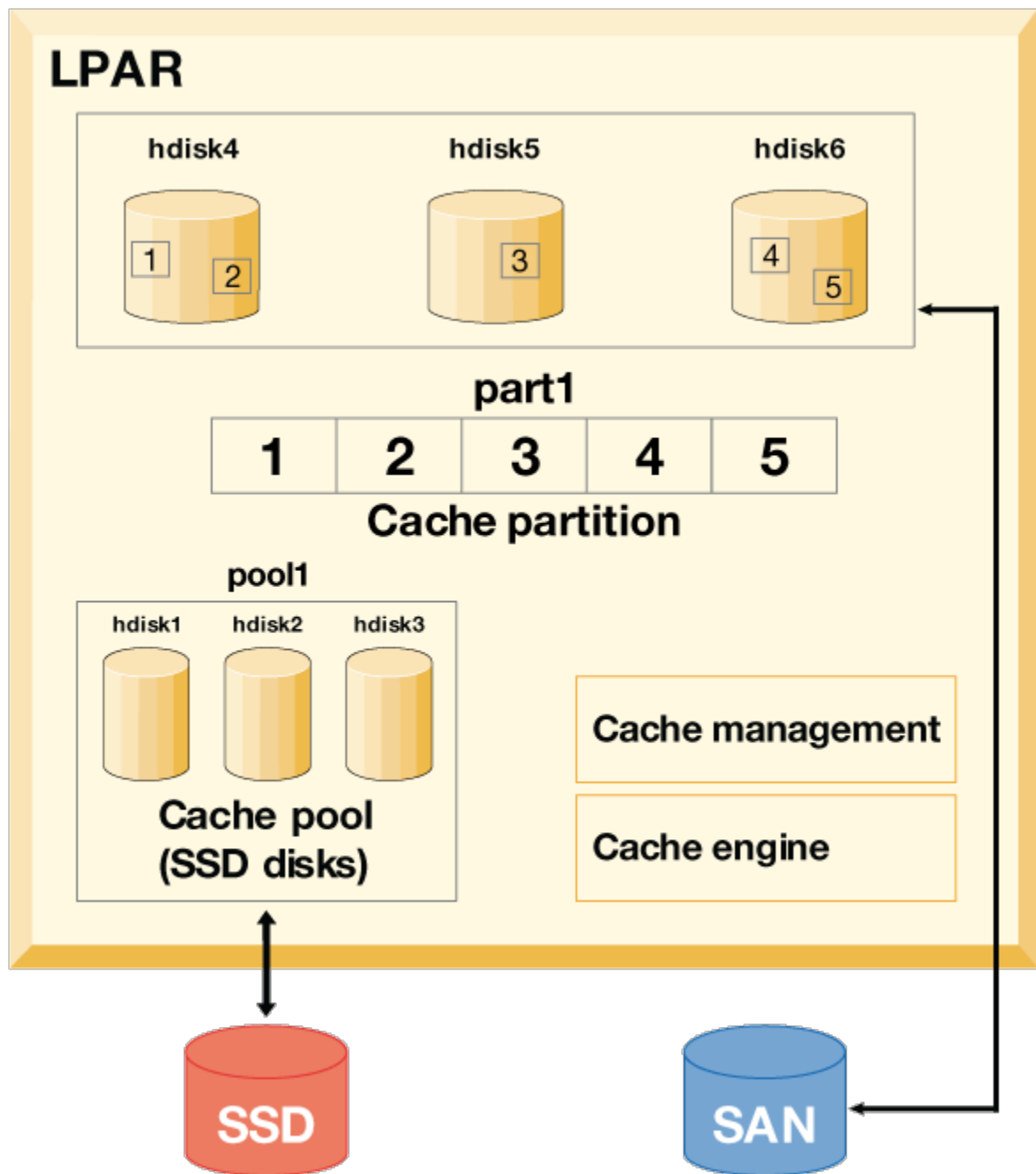


Figure 16. Storage data caching: Configuration for dedicated cache device

Consider the cache devices are **hdisk1**, **hdisk2**, and **hdisk3**, and the target devices are **hdisk4**, **hdisk5**, and **hdisk6**. To start and monitor the caching of the target devices, complete the following steps:

1. Create a cache pool on the SSD or Virtual Persistent Memory (vPMEM) storage.

```
# cache_mgt pool create -d hdisk1,hdisk2,hdisk3 -p cmpool0
```

2. Create a cache partition of 80 MB from the cache pool.

```
# cache_mgt partition create -p cmpool0 -s 80M -P part1
```

3. Assign the cache partition to the target disks that you want to cache.

```
# cache_mgt partition assign -t hdisk4 -P part1
# cache_mgt partition assign -t hdisk5 -P part1
# cache_mgt partition assign -t hdisk6 -P part1
```

4. Start caching of the target devices.

```
# cache_mgt cache start -t hdisk4
# cache_mgt cache start -t hdisk5
# cache_mgt cache start -t hdisk6
```

5. Monitor statistics on cache hits.

```
# cache_mgt monitor get -h -s
```

Caching storage data in virtual mode

In the virtual mode, the cache device is assigned to the Virtual I/O Server (VIOS).

In the virtual mode, the cache pool is created on the VIOS. The cache pool is then split into partitions on the VIOS. Each cache partition can be assigned to a virtual host (vhost) adapter. When the cache partition is discovered on the AIX logical partition (LPAR), the cache partition can be used for caching the target device. The cache partition can be migrated to another server because the cache device is virtual. Before the migration, the caching is automatically stopped on the source LPAR. As a part of the migration operation, a cache partition of the same size is created dynamically on the target VIOS if the caching software is installed and a cache pool is available on the target VIOS. During the migration, the cache partition is made available to the LPAR. When the migration is completed, caching is automatically started on the destination LPAR. In this case, the caching starts in an empty (unpopulated) state.

The following figure shows an example of caching configuration in virtual mode, where the cache device is on a VIOS LPAR and the target device is on an AIX LPAR.

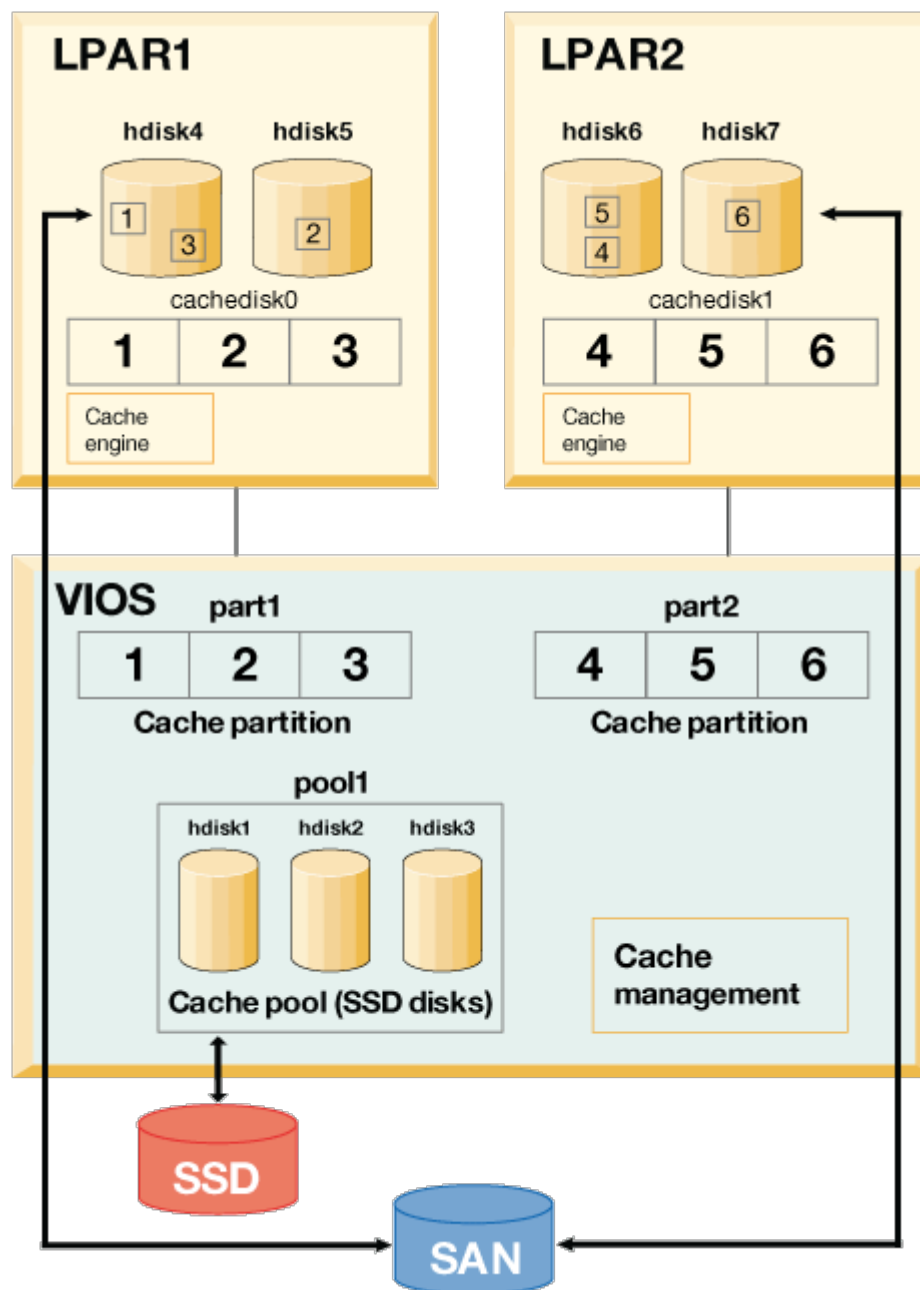


Figure 17. Storage data caching: Configuration for virtual cache device

Consider the cache devices are hdisk1, hdisk2, and hdisk3 (on VIOS LPAR), and the target device are hdisk4 and hdisk5 (on AIX LPAR). To start and monitor the caching of the target devices, complete the following procedure:

1. In the VIOS LPAR, create a cache pool by using the SSD storage.

```
# cache_mgt pool create -d hdisk1,hdisk2,hdisk3 -p cmpool0
```

2. In the VIOS LPAR, create a cache partition of 80 MB from the cache pool.

```
# cache_mgt partition create -p cmpool0 -s 80M -P part1
```

3. In the VIOS LPAR, assign the partition to a virtual host adapter.

```
# cache_mgt partition assign -P part1 -v vhost0
```

4. In the AIX LPAR, assign the cache partition to a target devices.

```
# cache_mgt partition assign -t hdisk4 -P cachedisk0  
# cache_mgt partition assign -t hdisk5 -P cachedisk0
```

5. In the AIX LPAR, start caching of the target devices.

```
# cache_mgt cache start -t hdisk4  
# cache_mgt cache start -t hdisk5
```

6. In the AIX LPAR, monitor statistics on cache hits.

```
# cache_mgt monitor get -h -s
```

Caching storage data in NPIV mode

In this mode, the cache device is available as a virtual Fibre Channel (N_Port ID Virtualization) device on the AIX logical partition (LPAR).

You must create a cache pool, and then a cache partition on the AIX LPAR. Only one cache partition can be created on the AIX LPAR. You can use the cache partition to cache any number of target devices on the AIX LPAR. The LPAR can be migrated to another server because the cache device is available from the storage area network (SAN). The cache device must be made visible on the destination system. The caching operation can continue during the migration process, and the cache will be populated after the migration completes.

Caching the target device in NPIV mode is same as caching the storage data in dedicated mode except that the cache device is available from the SAN. However, the procedure to cache the target device is same as caching the storage data in dedicated mode.

Managing storage data cache

Although, the cache is configured, the caching requirements might change over time. You might want to add new workloads to be cached. To fulfill the changing requirements, the cache pool can be extended with additional cache devices, a new cache partition can be created in an existing pool, or an existing partition can be increased in size.

You can use the following examples to manage the caching configuration:

- To add a cache device to the pool, enter the following command:

```
# cache_mgt pool extend -p pool1 -d hdisk4 -f
```

The **-f** flag overrides any existing usage of the disk (hdisk4) if the disk contains an existing volume group.

- To create a partition for a new workload of size 100 MB, enter the following command:

```
# cache_mgt partition create -p pool1 -s 100M -P part2
```

- To increase the size of an existing partition by 20 MB, enter the following command:

```
# cache_mgt partition extend -p part1 -s 20M
```

High-availability considerations

If the target devices, which are cached, are part of a resource group that is managed in a high-availability cluster, the failover operation must be planned properly.

Caching can be activated only on one node at a time. Before any failover event is initiated, you must ensure that the caching operation is disabled on the original system. After the failover to an alternative system is completed, you must manually enable the caching software.

To enable the caching software, complete the following steps:

1. Stop the caching on the original system.

```
# cache_mgt cache stop -t hdisk2
```

2. Start the caching on the new system after the failure recovery is completed.

```
# cache_mgt cache start -t hdisk2
```

Monitoring cache statistics

Cache statistics for each target device can be displayed by using the **cache_mgt monitor get** command.

For example, if hdisk1 is the only target device that is cached, the output of the **cache_mgt** command might be similar to the following example:

```
# cache_mgt monitor get -h -s
ETS Device I/O Statistics -- hdisk1
Start time of Statistics -- Mon Mar 27 07:10:41 2017
-----
Read Count:                152125803
Write Count:                79353626
Read Hit Count:             871
Partial Read Hit Count:     63
Read Bytes Xfer:            10963365477376
Write Bytes Xfer:           4506245999616
Read Hit Bytes Xfer:        48398336
Partial Read Hit Bytes Xfer: 5768192
Promote Read Count:         2033078104
Promote Read Bytes Xfer:    532959226494976
```

The **cache_mgt** command provides the following caching metrics:

Read Count

The total number of all the read operations that are issued by all applications to the target device. This value also includes the read operations to the cache device, if the data is available in the cache device. This value is the total number of separate read requests and does not indicate the size (byte count) of the read requests.

Write Count

The total number of write operations that are issued to the target device. This value is the total number of separate write requests and does not indicate the size (byte count) of the write requests.

Read Hit Count

The total number of read operations that were issued to the target device that are *full read hits*. A *full read hit* is a read operation instance in which a read request is fulfilled completely by the cache device. The **Read Hit Count** value is the total number of separate read hit requests and does not indicate the size (byte count) of the read requests. This value is included in the **Read Count** value.

Partial Read Hit Count

The total number of read operations that were issued to the target device that are *partial read hits*. A *partial read hit* is a read operation instance in which a read request is fulfilled partially by the cache device. The remaining data, which is not available in the cache device, must be acquired from the target device. The **Partial Read Hit Count** value is the total count of separate read requests and does not indicate the size (byte count) of the read requests. This value is included in the **Read Count** value.

Read Bytes Xfer

The total number of bytes that are transferred in all read requests that were issued from applications to the target device. This value represents the total byte count of *full read hit* data, *partial read hit* data, and any data that is acquired from the target device.

Write Bytes Xfer

The total number of bytes that are transferred in all write requests that were issued from applications to the target device.

Read Hit Bytes Xfer

The total number of bytes that are read during *full read hit* instances.

Partial Read Hit Bytes Xfer

The total number of bytes that are read during *partial read hit* instances.

Promote Read Count

The total number of read commands that are issued to the target device when the data is populated into the cache device. This value does not indicate the number of instances when the data is populated into the cache device because a single request to populate data into the cache device might be divided into multiple read operations if the maximum transfer size of data to the target disk is less than the request size.

Promote Read Bytes Xfer

The total number of bytes that are read from the target device when data is populated to the cache device.

Login names, system IDs, and passwords

The operating system must know who you are in order to provide you with the correct environment.

To identify yourself to the operating system, log in by typing your *login name* (also known as your user ID or user name) and a *password*. Passwords are a form of security. People who know your login name cannot log in to your system unless they know your password.

If your system is set up as a multiuser system, each authorized user will have an account, password, and login name on the system. The operating system keeps track of the resources used by each user. This is known as *system accounting*. Each user will be given a private area in the storage space of the system, called the *file system*. When you log in, the file system appears to contain only your files, although there are thousands of other files on the system.

It is possible to have more than one valid login name on a system. If you want to change from one login name to another, you do not have to log out of the system. Rather, you can use the different login names simultaneously in different shells or consecutively in the same shell without logging out. In addition, if your system is part of a network with connections to other systems, you can log in to any of the other systems where you have a login name. This is referred to as a *remote login*.

When you have finished working on the operating system, log out to ensure that your files and data are secure.

Logging in to the operating system

To use the operating system, your system must be running, and you must be logged in. When you log in to the operating system, you identify yourself to the system and allow the system to set up your environment.

Your system might be set up so that you can only log in during certain hours of the day and on certain days of the week. If you attempt to log in at a time other than the time allowed, access will be denied. Your system administrator can verify your login times.

You log in at the login prompt. When you log in to the operating system, you are automatically placed into your home directory (also called your *login directory*).

After your system is turned on, log in to the system to start a session.

1. Type your login name following the **login:** prompt:

```
login: LoginName
```

For example, if your login name is denise:

```
login: denise
```

2. If the **password:** prompt is displayed, type your password.

(The screen does not display your password as you type it.)

```
password: [your password]
```

If the password prompt is not displayed, you have no password defined; you can begin working in the operating system.

If your machine is not turned on, do the following before you log in:

1. Set the power switches of each attached device to On.
2. Start the system unit by setting the power switch to On (I).
3. Look at the three-digit display. When the self-tests complete without error, the three-digit display is blank.

If an error requiring attention occurs, a three-digit code remains, and the system unit stops. See your system administrator for information about error codes and recovery.

When the self-tests complete successfully, a login prompt similar to the following is displayed on your screen:

```
login:
```

After you have logged in, depending on how your operating system is set up, your system will start up in either a command line interface (shell) or a graphical interface (for example, AIXwindows or Common Desktop Environment (CDE)).

If you have questions concerning the configuration of your password or user name, please consult your system administrator.

Logging in more than one time (login command)

If you are working on more than one project and want to maintain separate accounts, you can have more than one concurrent login. You do this by using the same login name or by using different login names to log in to your system.

Note: Each system has a maximum number of login names that can be active at any given time. This number is determined by your license agreement and varies between installations.

For example, if you are already logged on as `denise1` and your other login name is `denise2`, at the prompt, type the following:

```
login denise2
```

If the **password:** prompt is displayed, type your password. (The screen does not display your password as you type it.) You now have two logins running on your system.

Becoming another user on a system (su command)

You can change the user ID associated with a session (if you know that user's login name) by using the **su** (switch user) command.

For example, if you want to switch to become user `joyce`, at the prompt, type the following:

```
su joyce
```

If the **password:** prompt is displayed, type the password for user `joyce`. Your user ID is now `joyce`. If you do not know the password, the request is denied.

To verify that your user ID is `joyce`, use the **id** command.

Suppressing login messages

After a successful login, the **login** command displays the message of the day, the date and time of the last successful and unsuccessful login attempts for this user, and the total number of unsuccessful login attempts for this user since the last change of authentication information (usually a password). You can suppress these messages by including a `.hushlogin` file in your home directory.

At the prompt in your home directory, type the following command:

```
touch .hushlogin
```

The **touch** command creates the empty file named `.hushlogin` if it does not exist. The next time you log in, all login messages will be suppressed. You can instruct the system to retain only the message of the day and suppress other login messages.

Logging out of the operating system (exit and logout commands)

To log out of the operating system, do one of the following at the system prompt.

Press the end-of-file control-key sequence (Ctrl-D keys).

OR

Type `exit`.

OR

Type `logout`.

After you log out, the system displays the **login:** prompt.

Displaying user IDs

To display the system identifications (IDs) for a specified user, use the **id** command. The system IDs are numbers that identify users and user groups to the system.

The **id** command displays the following information, when applicable:

- User name and real user ID
- Name of the user's group and real group ID
- Name of the user's supplementary groups and supplementary group IDs, if any

For example, at the prompt, type the following:

```
id
```

The system displays information similar to the following:

```
uid=1544(sah) gid=300(build) euid=0(root) egid=9(printq) groups=0(system),10(audit)
```

In this example, the user has user name `sah` with an ID number of 1544; a primary group name of `build` with an ID number of 300; an effective user name of `root` with an ID number of 0; an effective group name of `printq` with an ID number of 9; and two supplementary group names of `system` and `audit`, with ID numbers 0 and 10, respectively.

For example, at the prompt, type the following:

```
id denise
```

The system displays information similar to the following:

```
uid=2988(denise) gid=1(staff)
```

In this example, the user denise has an ID number of 2988 and has only a primary group name of staff with an ID number of 1.

Displaying your login name (whoami and logname commands)

When you have more than one concurrent login, it is easy to lose track of the login names or, in particular, the login name that you are currently using. You can use the **whoami** and **logname** commands to display this information.

Using the whoami command

To determine which login name is being used, at the prompt, type the following:

```
whoami
```

The system displays information similar to the following:

```
denise
```

In this example, the login name being used is denise.

Using the who am i command

A variation of the **who** command, the **who am i** command, allows you to display the login name, terminal name, and time of the login. At the prompt, type the following:

```
who am i
```

The system displays information similar to the following:

```
denise pts/0 Jun 21 07:53
```

In this example, the login name is denise, the name of the terminal is pts/0, and this user logged in at 7:53 a.m. on June 21.

Using the logname command

Another variation of the **who** command, the **logname** command displays the same information as the **who** command.

At the prompt, type the following:

```
logname
```

The system displays information similar to the following:

```
denise
```

In this example, the login name is denise.

Displaying the operating system name (uname command)

To display the name of the operating system, use the **uname** command.

For example, at the prompt, type the following:

```
uname
```

The system displays information similar to the following:

```
AIX
```

In this example, the operating system name is AIX.

Displaying your system name (uname command)

To display the name of your system if you are on a network, use the `uname` command with the `-n` flag. Your system name identifies your system to the network; it is not the same as your login ID.

For example, at the prompt, type the following:

```
uname -n
```

The system displays information similar to the following:

```
barnard
```

In this example, the system name is barnard.

Displaying all users who are logged in

To display information about all users currently logged into the local system, use the `who` command.

The following information is displayed: login name, system name, and date and time of login.

Note: This command only identifies logged-in users on the local node.

To display information about who is using the local system node, type the following:

```
who
```

The system displays information similar to the following:

```
joe    1ft/0 Jun 8 08:34
denise pts/1 Jun 8 07:07
```

In this example, the user `joe`, on terminal `1ft/0`, logged in at 8:34 a.m. on June 8.

See the `who` command.

Passwords

A unique password provides some system security for your files.

Your system associates a password with each account. Security is an important part of computer systems because it keeps unauthorized people from gaining access to the system and from tampering with other users' files. Security can also allow some users exclusive privileges to which commands they can use and which files they can access. For protection, some system administrators permit the users access only to certain commands or files.

Password guidelines

You should have a unique password. *Passwords should not be shared.* Protect passwords as you would any other company asset. When creating passwords, make sure they are difficult to guess, but not so difficult that you have to write them down to remember them.

Using obscure passwords keeps your user ID secure. Passwords based on personal information, such as your name or birthday, are poor passwords. Even common words can be easily guessed.

Good passwords have at least six characters and include nonalphabetic characters. Strange word combinations and words purposely misspelled are also good choices.

Note: If your password is so hard to remember that you have to write it down, it is not a good password.

Use the following guidelines when selecting a password:

- Do not use your user ID as a password. Do not use it reversed, doubled, or otherwise modified.
- Do not reuse passwords. The system might be set up to deny the reuse of passwords.
- Do not use any person's name as your password.

- Do not use words that can be found in the online spelling-check dictionary as your password.
- Do not use passwords shorter than six characters.
- Do not use obscene words; they are some of the first ones checked when guessing passwords.
- Do use passwords that are easy to remember, so you won't have to write them down.
- Do use passwords that use both letters and numbers and that have both lowercase and uppercase letters.
- Do use two words, separated by a number, as a password.
- Do use pronounceable passwords. They are easier to remember.
- Do not write passwords down. However, if you must write them down, place them in a physically secure place, such as a locked cabinet.

Changing passwords (passwd command)

To change your password, use the **passwd** command.

1. At the prompt, type the following:

```
passwd
```

If you do not already have a password, skip step 2.

2. The following prompt displays:

```
Changing password for UserID
UserID's Old password:
```

This request keeps an unauthorized user from changing your password while you are away from your system. Type your current password, and press Enter.

3. The following prompt is displayed:

```
UserID's New password:
```

Type the new password you want, and press Enter.

4. The following prompt is displayed, asking you to re-type your new password.

```
Enter the new password again:
```

This request protects you from setting your password to a mistyped string that you cannot re-create.

Password nullification (passwd command)

If you do not want to type a password each time you log in, set your password to null (blank).

To set your password to null, type the following:

```
passwd
```

When you are prompted for the new password, press Enter or Ctrl-D.

The **passwd** command does not prompt again for a password entry. A message verifying the null password displays.

See the **passwd** command for more information and the complete syntax.

Command summary for login names, system IDs, and passwords

Commands are available for working with login names, system IDs, and passwords.

Login and logout commands

Item	Description
<u>login</u>	Initiates your session
<u>logout</u>	Stops all your processes
<u>shutdown</u>	Ends system operation
<u>su</u>	Changes the user ID associated with a session
<u>touch</u>	Updates the access and modification times of a file, or creates an empty file

User and system identification commands

Item	Description
<u>id</u>	Displays the system identifications of a specified user
<u>logname</u>	Displays login name.
<u>uname</u>	Displays the name of the current operating system
<u>who</u>	Identifies the users currently logged in
<u>whoami</u>	Displays your login name

Password command

Item	Description
<u>passwd</u>	Changes a user's password

Common Desktop Environment

With the Common Desktop Environment (CDE), you can access networked devices and tools without having to be aware of their location. You can exchange data across applications by simply dragging and dropping objects.

Many tasks that previously required complex command line syntax can be done more easily and similarly from platform to platform. For example, you can centrally configure and distribute applications to users. You can also centrally manage the security, availability, and interoperability of applications for the users you support.

Note: The Common Desktop Environment (CDE) 1.0 Help volumes, Web-based documentation, and hardcopy manuals may refer to the desktop as the Common Desktop Environment, the AIXwindows desktop, CDE 1.0, or the desktop.

Enabling and disabling desktop autostart

You might find it more convenient to set up your system to start Common Desktop Environment automatically when the system is turned on.

You can do this through the System Management Interface Tool (SMIT) or from the command line.

Prerequisites

You must have root user authority to enable or disable desktop autostart.

Consult the following table to determine how to enable or disable desktop autostart.

Starting and Stopping the Common Desktop Environment Automatically		
Task	SMIT Fast Path	Command or File
Enabling the desktop autostart ¹	smit dtconfig	/usr/dt/bin/dtconfig -e
Disabling the desktop autostart ¹	smit dtconfig	/usr/dt/bin/dtconfig -d

Note: Restart the system after completing this task.

Starting the Common Desktop Environment manually

Use this procedure to start the Common Desktop Environment manually.

1. Log in to your system as root.
2. At the command line, type the following:

```
/usr/dt/bin/dtlogin -daemon
```

A **Desktop Login** screen is displayed. When you log in, a desktop session starts.

Stopping the Common Desktop Environment manually

When you manually stop the Login Manager, all X Servers and desktop sessions that the Login Manager started are stopped.

1. Open a terminal emulator window, and log in as root.
2. Obtain the process ID of the Login Manager by typing the following:

```
cat /var/dt/Xpid
```

3. Stop the Login Manager by typing:

```
kill -term process_id
```

Modifying desktop profile

When you log in to the desktop, the shell environment file (.profile or .login) is not automatically read. The desktop runs the X Server before you log in, so the function provided by the .profile file or the .login file must be provided by the desktop's Login Manager.

User-specific environment variables are set in */Home Directory/.dtprofile*. A template for this file is located in */usr/dt/config/sys.dtprofile*. Place variables and shell commands in .dtprofile that apply only to the desktop. Add lines to the end of the .dtprofile to incorporate the shell environment file.

System-wide environment variables can be set in the Login Manager configuration files. For details on configuring environment variables, see the *Common Desktop Environment 1.0: Advanced User's and System Administrator's Guide*.

Adding and removing displays and terminals for the Common Desktop Environment

You can add and remove displays and terminals for the Common Desktop Environment.

The Login Manager can be started from a system with a single local bitmap or graphics console. Many other situations are also possible, however (see the following figure). You can start the Common Desktop Environment from:

- Local consoles
- Remote consoles

- Bitmap and character-display X terminal systems running on a host system on the network

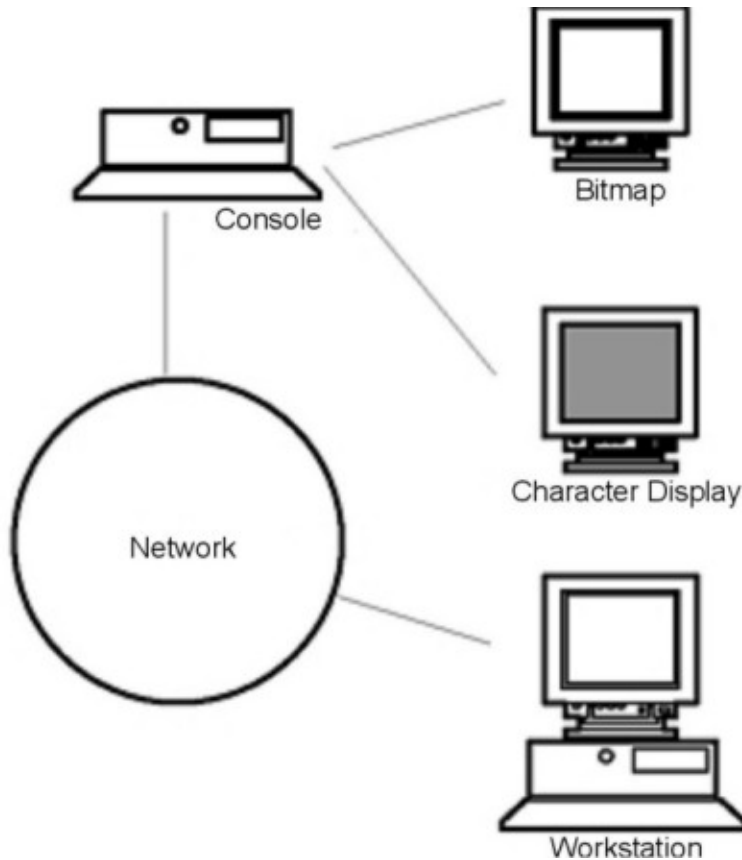


Figure 18. CDE Interface Points

An X terminal system consists of a display device, keyboard, and mouse that runs only the X Server. Clients, including the Common Desktop Environment, are run on one or more host systems on the networks. Output from the clients is directed to the X terminal display.

The following Login Manager configuration tasks support many possible configurations:

- Removing a local display
- Adding an ASCII or character-display terminal

To use a workstation as an X terminal, type the following at a command line:

```
/usr/bin/X11/X -query hostname
```

The X Server of the workstation acting as an X terminal must:

- Support XDMCP and the **-query** command-line option.
- Provide xhost permission (in `/etc/X*.hosts`) to the terminal host.

To remove a local display, remove its entry in the `Xservers` file in the `/usr/dt/config` directory.

A *character-display terminal*, or *ASCII terminal*, is a configuration in which the terminal is not a bitmap device.

To add an ASCII or character-display console if no bitmap display is present, perform the following steps:

1. If the `/etc/dt/config/Xservers` file does not exist, copy the `/usr/dt/config/Xservers` file to the `/etc/dt/config` directory.
2. If you have to copy the `Xservers` file to `/etc/dt/config`, change or add the `Dtlogin.servers:` line in `/etc/dt/config/Xconfig` to:


```
Dtlogin*servers: /etc/dt/config/Xservers
```

3. Comment out the line in `/etc/dt/config/Xservers` that starts the X Server.

```
# * Local local@console /path/X :0
```

This disables the **Login Option Menu**.

4. Read the Login Manager configuration files.

To add a character-display console if a bitmap display exists, perform the following steps:

1. If the `/etc/dt/config/Xservers` file does not exist, copy the `/usr/dt/config/Xservers` file to the `/etc/dt/config` directory.
2. If you have to copy the `Xservers` file to `/etc/dt/config`, change or add the `Dtlogin.servers:` line in `/etc/dt/config/Xconfig` to:

```
Dtlogin*servers: /etc/dt/config/Xservers
```

3. Edit the line in `/etc/dt/config/Xservers` that starts the X Server to:

```
* Local local@none /path/X :0
```

4. Read the Login Manager configuration files.

Displaying device customization for the Common Desktop Environment

You can configure the Common Desktop Environment Login Manager to run on systems with two or more display devices.

When a system includes multiple displays, the following configuration requirements must be met:

- A server must be started on each display.
- No Windows Mode must be configured for each display.

It might be necessary or desirable to use different `dtlogin` resources for each display.

It might also be necessary or desirable to use different system-wide environment variables for each display device.

Starting the server on each display device

Start a server on each display device using this procedure.

1. If the `/etc/dt/config/Xservers` file does not exist, copy the `/usr/dt/config/Xservers` file to the `/etc/dt/config` directory.
2. If you have to copy the `Xservers` file to `/etc/dt/config`, change the `Dtlogin.servers:` line in `/etc/dt/config/Xconfig` to:

```
Dtlogin*servers: /etc/dt/config/Xservers
```

3. Edit `/etc/dt/config/Xservers` to start an X Server on each display device.

The general syntax for starting the server is:

```
DisplayName DisplayClass DisplayType [ @ite ] Command
```

Only displays with an associated Internal Terminal Emulator (ITE) can operate in *No Windows Mode*. No Windows Mode temporarily disables the desktop for the display and runs a *getty* process if one is not already started. A *getty* process is a UNIX program that sets terminal type and is used in the login process.

This allows you to log in and perform tasks not possible under the Common Desktop Environment. When you log out, the desktop is restarted for the display device. If a *getty* process is not already running on a display device, Login Manager starts one when No Windows Mode is initiated.

In the default configuration, when `ite` is omitted, `display:0` is associated with the ITE (`/dev/console`).

Setting up a different display as ITE

Use this procedure to set up a different display as ITE.

To specify a different display as ITE:

- On the ITE display, set ITE to the character device.
- On all other displays, set ITE to none.

The following examples show entries in the `Xserver` file which start a server on three local displays on `sysaaa:0`. Display `:0` will be the console (ITE).

```
sysaaa:0 Local local /usr/bin/X11/X :0
sysaaa:1 Local local /usr/bin/X11/X :1
sysaaa:2 Local local /usr/bin/X11/X :2
```

On host `sysbbb`, the bitmap display `:0` is not the ITE; the ITE is associated with device `/dev/ttyi1`. The following entries in the `Xserver` file start servers on the two bitmap displays with No Windows Mode enabled on `:1`.

```
sysaaa:0 Local local@none /usr/bin/X11/X :0
sysaaa:1 Local local@ttyi1 /usr/bin/X11/X :1
```

Setting up the display name in Xconfig

You cannot use regular `hostname:0` syntax for the display name in `/etc/dt/config/Xconfig`.

To specify the display name in `Xconfig`:

- Use an underscore in place of the colon.
- In a fully qualified host name, use underscores in place of the periods.

The following example shows the setup of the display name in `Xconfig`:

```
Dtlogin.claaa_0.resource: value
Dtlogin.sysaaa_prsm_ld_edu_0.resource: value
```

Using different Login Manager resources for each display

To use different Login Manager resources for each display, perform the following steps:

1. If the `/etc/dt/config/Xconfig` file does not exist, copy the `/usr/dt/config/Xconfig` file to the `/etc/dt/config` directory.
2. Edit the `/etc/dt/config/Xconfig` file to specify a different resource file for each display.
For example:

```
Dtlogin.DisplayName.resources: path/file
```

where *path* is the path name of the Xresource files to be used, and *file* is the file name of the Xresource files to be used.

3. Create each of the resource files specified in the `Xconfig` file.
A language-specific Xresources file is installed in `/usr/dt/config/<LANG>`.
4. In each file, place the `dtlogin` resources for that display.

The following example shows lines in the `Xconfig` file which specify different resource files for three displays:

```
Dtlogin.sysaaa_0.resources: /etc/dt/config/Xresources0
Dtlogin.sysaaa_1.resources: /etc/dt/config/Xresources1
Dtlogin.sysaaa_2.resources: /etc/dt/config/Xresources2
```

Running different scripts for each display

Use this procedure to run a particular script for a specific display.

1. If the `/etc/dt/config/Xconfig` file does not exist, copy the `/usr/dt/config/Xconfig` file to the `/etc/dt/config` directory.
2. Use the startup, reset, and setup resources in `/etc/dt/config/Xconfig` to specify different scripts for each display (these files are run instead of the `Xstartup`, `Xreset`, and `Xsetup` files):

```
Dtlogin*DisplayName*startup: /path/file
Dtlogin*DisplayName*reset: /path/file
Dtlogin*DisplayName*setup: /path/file
```

where *path* is the path name of the file to be used, and *file* is the file name of the file to be used. The startup script is run as root after the user has logged in, before the Common Desktop Environment session is started.

The script `/usr/dt/config/Xreset` can be used to reverse the setting made in the `Xstartup` file. The `Xreset` file runs when the user logs out.

The following example shows lines in the `Xconfig` file which specify different scripts for two displays:

```
Dtlogin.sysaaa_0*startup: /etc/dt/config/Xstartup0
Dtlogin.sysaaa_1*startup: /etc/dt/config/Xstartup1
Dtlogin.sysaaa_0*setup: /etc/dt/config/Xsetup0
Dtlogin.sysaaa_1*setup: /etc/dt/config/Xsetup1
Dtlogin.sysaaa_0*reset: /etc/dt/config/Xreset0
Dtlogin.sysaaa_1*reset: /etc/dt/config/Xreset1
```

Setting different system-wide environment variables for each display

Use this procedure to customize system-wide environment variables to each display.

To set different system-wide environment variables for each display:

1. If the `/etc/dt/config/Xconfig` file does not exist, copy the `/usr/dt/config/Xconfig` file to the `/etc/dt/config` directory.
2. Set the environment resource in `/etc/dt/config/Xconfig` separately for each display:

```
Dtlogin*DisplayName*environment: value
```

The following rules apply to environment variables for each display:

- Separate variable assignments with a space or tab.
- Do not use the environment resource to set TZ and LANG.
- There is no shell processing within the `Xconfig` file.

The following example shows lines in the `Xconfig` file which set variables for two displays:

```
Dtlogin*syshere_0*environment:EDITOR=vi SB_DISPLAY_ADDR=0xB000000
Dtlogin*syshere_1*environment:EDITOR=emacs \
SB_DISPLAY_ADDR=0xB000000
```

Live Partition Mobility with Host Ethernet Adapters

Using the Live Partition Mobility (LPM) with Host Ethernet Adapters (HEA) feature of IBM PowerVM® software, you can migrate an AIX LPAR and hosted applications from one physical partition to another physical partition while HEA is assigned to the migration partition.

During the migration, HEA will be removed from the migrating partition, and HEA will not be restored on the partitions when migration is complete. However, your network connectivity will remain unaffected.

Requirements for Live Partition Mobility with HEA

Before you can start using LPM with HEA, you must make sure that your system environment meets configuration and access requirements.

Partition requirements

- The CEC source and CEC target must be capable for partition migration.
- The source AIX LPAR must not have any physical resources with the setting Required in its profile.
- The source AIX LPAR must not have any physical resources besides a HEA.

Access requirements

- You must have root authority on the partition that you want to migrate.
- You must have hscroot authority or equivalent authority required for partition migration on the source HMC and destination HMC.

Configuration requirements

- HEA must not have the Required setting in the partition profile, but it can have the Desired setting in the profile.
- All HEA must be configured under EtherChannel as primary adapters.
- All primary adapters in EtherChannel must be HEA.
- The backup adapter for EtherChannel must be a Virtual Ethernet adapter.
- A minimum of one EtherChannel must be configured with HEA as a primary adapter and Virtual Ethernet adapter as a backup adapter.
- A maximum of four EtherChannel is supported.
- EtherChannel failover must be functional.
- You must verify that both the source system and target system are set up for partition migration.
- If you are migrating between two HMCs, you must set up SSH authentication between the source HMC and remote HMC. You must run the **mkauthkeys** command on the source HMC before you start the migration.

Running Live Partition Mobility with HEA

You can run LPM with HEA using the SMIT interface.

Review the [“Requirements for Live Partition Mobility with HEA ” on page 236](#) topic before you attempt to use LPM with HEA.

To complete an LPM with HEA partition migration, complete the following steps:

1. From the command prompt, enter the following SMIT fastpath: **smitty migration** to display the Live Partition Mobility with Host Ethernet Adapter (HEA) menu.
2. Specify the source HMC hostname or IP address.
3. Specify the source HMC username.
4. Enter **no** if source and destination systems are managed by same HMC and go to step 5. Enter **yes** if the source and destination systems are managed by different HMCs and complete the following steps:

Note: You must run the **mkauthkeys** command on the source HMC before you enter **yes**.

- a) Specify the remote HMC hostname or IP address.
 - b) Specify the source HMC username.
5. Specify the name of the source system.
 6. Specify the name of the destination system.
 7. Specify the name of the partition you want to migrate.

8. Enter **no** if you want to perform the migration without validation. Enter **yes** if you want to perform migration validation only. If you specify **yes** the migration is not performed, and only the validation is performed.

Note: You should perform partition migration validation before performing the partition migration.

9. Verify that all fields have the correct information and press **Enter** to execute the migration.

Note: You will be prompted two times to enter the password. Enter the password for the source HMC username that you previously specified in step 4b.

In this example partition X is migrating from the CEC C that is managed by HMC A to the CEC D that is managed by HMC B. Run the **mkauthkeys** command on HMC A for authentication between HMC A and HMC B.

For this migration process, the following values are specified in SMIT:

```
Source HMC Hostname or IP address: A
Source HMC username: hscroot
Migration between two HMCs: yes
  Remote HMC hostname or IP address: B
  Remote HMC username: hscroot
Source system: C
Destination system: D
Migrating Partition name: X
Migration validation only: no
```

Another example would be if partition X is migrating from CEC C managed by HMC A to CEC D that is also managed by HMC A.

For this migration process, the following values are specified in SMIT:

```
Source HMC Hostname or IP address: A
Source HMC username: hscroot
Migration between two HMCs: no
  Remote HMC hostname or IP address:
  Remote HMC username:
Source system: C
Destination system: D
Migrating Partition name: X
Migration validation only: no
```

In case of migration failure, follow the procedure to perform Live Partition Mobility from the source HMC.

Migrating a NIM client by using LPM

When Live Partition Mobility (LPM) is used to move a machine from one physical server to another and the machine is defined as a Network Installation Management (NIM) client, the NIM administrator must update the *cpuid* attribute for the NIM client to reflect the new hardware value after the LPM migration completes.

To update the *cpuid* attribute, complete the following steps:

1. On the NIM client, acquire the new cpuid ID by running the following command:

```
uname -a
```

2. On the NIM master, run the following command:

```
nim -o change -a cpuid=cpuid client
```

Note: The `OS_install` network installer no longer supports the installation of the Linux® operating system because of the removal of Cluster Systems Management (CSM) support in the AIX operating system.

Relocating an adapter for DLPAR

You must configure the graphic adapter before relocating an adapter for dynamic logical partitioning (DLPAR) operations.

Use the following instructions to dynamically relocate a graphics adapter, such as FC 5748:

1. Ensure that no current processes (Desktop, and Xserver) are using the graphics adapter (for example, /dev/lft0).
2. Verify that the console is not set to the lft0. To identify the defined or available dependent interfaces (lft or rcm), enter the following command:

```
lsdev -C | grep lft
lsdev -C | grep rcm
```

To obtain a parent Peripheral Component Interconnect (PCI) adapter after the graphics adapter is defined, enter the following command:

```
odmget -q name=<cortina adapter name. for instance cor0> CuDv
```

This command provides information about the parent and cortina.

3. To remove all dependent interfaces to make the adapter ready for DLPAR operation, enter the following command:

```
# pdisable lft0

# rmdev -l rcm0
rcm0 Defined

# rmdev -l lft0
lft0 Defined

# rmdev -Rdl pci23
cor0 deleted
pci23 deleted
```

The management console interface can be used for DLPAR operations on the graphics adapter.

Loopback device

A loopback device is a device that can be used as a block device to access files.

The loopback file can contain an ISO image, a disk image, a file system, or a logical volume image. For example, by attaching a CD-ROM ISO image to a loopback device and mounting it, you can access the image the same way that you can access the CD-ROM device.

Use the **loopmount** command to create a loopback device, to bind a specified file to the loopback device, and to mount the loopback device. Use the **loopumount** command to unmount a previously mounted image file on a loopback device, and to remove the device. There is no limit on the number of loopback devices in AIX. A loopback device is never created by default; you must explicitly create the device. The block size of a loopback device is always 512 bytes.

A new device can also be created with the **mkdev** command, changed with the **chdev** command, and removed with the **rmdev** command. After a device is created, it can be either mounted to access the underlying image or used as a block device for raw I/O. Information about the underlying image can be retrieved with the **ioctl (IOCIINFO)** command.

The following restrictions apply to a loopback device in AIX:

- The **varyonvg** command on a disk image is not supported.
- A CD ISO, and DVD UDF+ISO, and other CD/DVD images are only supported in read-only format.
- An image file can be associated with only one loopback device.
- Loopback devices are not supported in workload partitions.

Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US*

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as the customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at [Copyright and trademark information at www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

Index

Special Characters

.hushlogin file [226](#)
/ (root) file system [76](#)
/export directory [81](#)
/opt file system [74](#)
/proc file system [74](#)
/usr/share directory [79](#)
/var file system [80](#)

A

absolute path name [120](#)
adapter location codes [168](#)
allocation group size [110](#)
allocations, file zero (kproc) [112](#)
API
 Workload Manager (WLM) [163](#)
application programming interface
 Workload Manager (API) [163](#)
ASCII terminals
 adding [231](#)
availability
 for adapter or power supply failure [55](#)
 for disk failure [54](#)

B

backup
 effect of fragments on [116](#)
blocks
 performance costs of [110](#)

C

cables
 checking connections [197](#)
caching
 statistics monitoring [223](#)
Caching
 Advantages [215](#)
 Components [217](#)
 Concept [215](#)
 Configuring [217](#)
 Configuring in dedicated mode [218](#)
 Configuring in NPIV mode [222](#)
 Configuring in virtual mode [220](#)
 High-availability considerations [222](#)
 Managing [222](#)
cd command [120, 122](#)
CD-ROM
 file systems [87](#)
CD-ROM File System (CDRFS) [105](#)
CDRFS file systems [87](#)
cfgmgr [199](#)
changing

 changing (*continued*)
 to another directory [122](#)
 to become another user [225](#)
character-display terminals
 adding [231](#)
chdev command [195](#)
checking file systems for inconsistencies [89](#)
command summaries
 login names [230](#)
 passwords [230](#)
 system IDs [230](#)
commands
 cd [120, 122](#)
 chdev [195](#)
 cp [123](#)
 df [83](#)
 dircmp [125](#)
 exit [226](#)
 id [225, 226](#)
 ln [119](#)
 login [225](#)
 logname [227](#)
 logout [226](#)
 lsattr [195](#)
 lsdev [195](#)
 mkdev [195](#)
 mkdir [121](#)
 mvdrr [121](#)
 passwd [229](#)
 pwd [122](#)
 rmdir [125](#)
 su [225](#)
 touch [226](#)
 uname [227, 228](#)
 who [227, 228](#)
 who am i [227](#)
 whoami [227](#)
commands and fastpaths [10](#)
Common Desktop Environment
 adding displays and terminals [231](#)
 customizing display devices [233](#)
 modifying profiles [231](#)
 removing displays and terminals [231](#)
cp command [123](#)
creating
 directories [121](#)
customizing
 display devices [233](#)

D

data compression
 fragments [107](#)
 performance costs of [114](#)
deleting
 directories [125](#)
device

- device (*continued*)
 - configuring a read/write optical drive [39](#)
 - installation [38](#)
- device configuration database
 - synchronizing with Logical Volume Manager [36](#)
- device drivers
 - effect of using fragments on size of [116](#)
- Device drivers [171](#)
- devices
 - changing attributes [195](#)
 - checking attributes [195](#)
 - checking the connections [197](#)
 - checking the ready state [198](#)
 - checking the software [195](#)
 - checking the state of [195](#)
 - classes [167](#)
 - configuring large numbers [39](#)
 - defining new [195](#)
 - location codes [168](#)
 - MPIO
 - cabling [179](#)
 - MPIO-capable [178](#)
 - nodes [166](#)
 - running diagnostics [198](#)
 - states [168](#)
- df command [83](#)
- diagnosing disk drive problems [27](#)
- dials/LPFKeys location codes [170](#)
- dircmp command [125](#)
- directories
 - changing [122](#)
 - comparing contents [125](#)
 - copying [123](#)
 - creating [121](#)
 - deleting [125](#)
 - displaying [122](#)
 - home [120](#)
 - mounting [101](#)
 - moving [121](#)
 - organization [120](#)
 - overview [119](#)
 - parent [120](#)
 - path names [120](#)
 - removing [125](#)
 - renaming [121](#)
 - root [119](#)
 - structure [120](#)
 - subdirectories [120](#)
 - types [119](#)
 - working [120](#)
- directory tree [120](#)
- disk
 - adding [12](#)
 - removing [49](#)
- disk drives
 - diagnosing [27](#)
 - freeing space on [27](#)
 - mounting space from another disk [28](#)
 - recovering from problems [27](#)
 - recovery of data
 - without reformatting [28](#)
 - removing obsolete files from [28](#)
 - restricting access to directories on [28](#)
 - unmounting file systems on a disk [85](#)
- disk drives (hard drives)
 - failure of
 - example of recovery from [32](#)
 - listing file systems [85](#)
- disk overflows, fixing [96](#)
- disk striping [62](#)
- disk utilization
 - effect of fragments on [107](#)
- diskless workstations
 - mount security [102](#)
- disks (hard drives)
 - configuring [19](#)
- displaying
 - available space [83](#)
 - file directory [122](#)
 - logged in users [228](#)
 - login name [227](#)
 - operating system name [227](#)
 - user ID [226](#)
 - your system name [228](#)
- DVD
 - file systems [87](#)

E

- EFS
 - encrypted file systems [82](#)
- enabled file systems
 - create [112](#)
 - free space [112](#)
 - large file geometry [112](#)
- enabling file systems
 - zero file allocations [112](#)
- encrypting logical volumes [53](#)
- Enhanced Journaled File System (JFS2) [105](#)
- error logging
 - checking for device errors [195](#)
- exclusive use RSET
 - exclusive use processor resource set [157](#)
- exit command [226](#)

F

- failed disk drive
 - example of recovery from [32](#)
- file system
 - bypassing [48](#)
 - images [116](#)
- file system fragment addressability [110](#)
- file system log [14](#)
- file systems
 - /opt [74](#)
 - /proc [74](#)
 - CD-ROM File System (CDRFS) [105](#)
 - CDRFS [87](#)
 - commands for managing [82](#), [84](#)
 - data compression [113](#)
 - description [224](#)
 - disk overflows [96](#)
 - Enhanced Journaled File System (JFS2) [105](#)
 - file tree
 - / (root) file system [76](#)
 - /export directory [81](#)

file systems (*continued*)

file tree (*continued*)

/usr file system [78](#)

/usr/share directory [79](#)

/var file system [80](#)

overview [74](#)

root (/) file system [76](#)

fixing damaged [94](#)

fragments [107](#)

groups

mounting [85](#)

unmounting [85](#)

home [74](#)

i-nodes [107](#)

Journal File System (JFS) [105](#)

large files [112](#)

management tasks [82](#)

mounting [85](#), [101](#)

Network File System (NFS) [105](#)

on read/write optical media [87](#)

reducing size in root volume group [90](#)

root [74](#)

space available [83](#)

sparse files [111](#)

structure [74](#)

types

CD-ROM [105](#)

DVD-ROM [105](#)

enhanced journaled file system (JFS2) [105](#)

journaled file system (JFS) [105](#)

network file system (NFS) [105](#)

UDFS [87](#)

unmounting [85](#)

verifying integrity of [89](#)

files

.hushlogin [226](#)

comparing [125](#)

mounting [101](#)

path names [120](#)

fixed-disk drives (hard drives)

also see disk drives [27](#)

fragments

and variable number of i-nodes [107](#)

effect on backup/restore [116](#)

effect on disk utilization [107](#)

limitation for device drivers [116](#)

performance costs of [110](#)

size of

identifying [109](#)

specifying [109](#)

H

hard disk [19](#)

home directory [120](#)

home file system [74](#)

hot disk removability [12](#), [49](#)

hot plug management

PCI [174](#)

hot removability [33](#), [49](#), [50](#)

hot spots in logical volumes [63](#)

I

i-node number [119](#)

i-nodes

and fragments [107](#)

number of bytes per (NBPI)

identifying [109](#)

specifying [109](#)

variable number of [109](#)

i-nodes, number of [110](#)

id command [225](#), [226](#)

importing user-defined volume groups [90](#)

index node reference number [119](#)

inter-disk allocation strategy [58](#)

intra-disk allocation strategy [61](#)

J

JFS

copy to another physical volume [117](#)

JFS (journaled file system)

data compression [113](#)

fragments [107](#)

maximum size of [110](#)

on read / write optical media [87](#)

size limitations [109](#)

with variable number of i-nodes [107](#)

JFS (journaled file system) log

size of [110](#)

JFS log [14](#)

JFS2 (enhanced journaled file system)

size limitations [109](#), [111](#)

JFS2 log [14](#)

Journal File System (JFS) [105](#)

L

limitations

logical volumes [35](#)

ln command [119](#)

location codes

adapter [168](#)

defined [168](#)

dials/LPFKeys [170](#)

multiprotocol port [171](#)

printer/plotter [169](#)

SCSI device [170](#)

tty [169](#)

logging in

as another user [225](#)

more than one time [225](#)

to the operating system [224](#)

logging out

of the operating system [226](#)

logical partitions

defining size of [90](#)

definition [44](#)

inter-disk allocation strategy [58](#)

logical volume

copy to another physical volume [13](#)

raw

define [48](#)

Logical Volume Manager [10](#)

- Logical Volume Manager (LVM)
 - definition [1](#)
 - synchronizing with device configuration database [36](#)
- logical volume storage
 - definition [41](#)
 - disk overflows [96](#)
 - file systems [45](#)
 - inter-disk allocation policy [58](#)
 - intra-disk allocation policy [61](#)
 - logical partitions [44](#)
 - logical volumes [44](#)
 - maximum sizes [45](#)
 - nonquorum volume groups [8](#)
 - physical volumes [42](#)
 - quorums [2](#)
 - volume groups [42](#)
 - write scheduling policy [56](#), [57](#)
- logical volumes
 - adding a file system on new [82](#)
 - changing name [13](#)
 - definition [44](#)
 - hot spots [63](#)
 - limitations [35](#)
 - map files [61](#)
 - moving contents to another system [17](#)
 - replacing a disk [34](#)
 - size
 - checking [82](#)
 - decreasing [82](#)
 - increasing [82](#)
 - strategy for [55](#)
 - striped [62](#)
 - volume group policy [64](#)
 - write-verify policy [62](#)
- logical-volume control block
 - not protected from raw-logical-volume access [48](#)
- login
 - directory [224](#)
 - displaying name [227](#)
 - name [224](#)
 - overview [224](#)
 - suppressing messages [226](#)
- login command [225](#)
- logname command [227](#)
- logout
 - overview [224](#)
- logout command [226](#)
- lsattr command [195](#)
- lsdev command [195](#)
- LVCB (logical-volume control block)
 - not protected from raw-logical-volume access [48](#)
- LVM [10](#), [37](#)

M

- maintenance [10](#)
- map files [61](#)
- Mirror Write Consistency (MWC) [57](#)
- mirroring
 - root volume group (rootvg) [46](#)
 - splitting a mirrored disk from a volume group [26](#)
 - volume group [46](#)

- mkdev command [195](#)
- mkdir command [121](#)
- modifying
 - desktop profiles [231](#)
- mount points [100](#)
- mounting
 - /etc/filesystem automatic mounts [101](#)
 - automatic mounts [101](#)
 - diskless workstation mounts
 - description of [104](#)
 - security [102](#)
 - file system mounting [101](#)
 - local
 - definition [101](#)
 - remote
 - definition [101](#)
 - using multiple mounts [101](#)
- MPIO
 - managing [178](#)
- Multi-path I/O [177](#)
- multiprotocol port
 - location codes [171](#)
- mvsdir command [121](#)

N

- NBPI [109](#)
- network
 - displaying system name [228](#)
- Network File System (NFS) [105](#)
- nonquorum volume groups [8](#)
- number of bytes per i-node (NBPI) [109](#)

O

- operating system
 - displaying name [227](#)
 - logging in [224](#)
 - logging out [226](#)
- optical drive
 - configuring [39](#)
- optical media
 - using file systems on read/write [87](#)

P

- paging space
 - allocating [65](#)
 - changing characteristics of [69](#)
 - changing size of hd6 [70](#)
 - characteristics for creating [67](#)
 - commands for managing [67](#)
 - early allocation mode [65](#)
 - late allocation mode [65](#)
 - moving hd6 [70](#)
 - overview [65](#)
 - removing [69](#)
- parent directory [120](#)
- passwd command [229](#)
- passwords
 - changing or setting [229](#)
 - description [224](#)
 - guidelines [228](#)

passwords (*continued*)
 setting to null [229](#)

path names

 absolute [120](#)
 directory [120](#)
 relative [120](#)

paths

 directory [120](#)

performance

 improving
 defining raw logical volumes [48](#)

physical partitions

 definition [43](#)
 size [43](#)

physical volume

 copy JFS to another [117](#)
 copy logical volume to another [13](#)

physical volumes

 configuring a disk [19](#)
 creating from available disk drive [20](#)
 definition [42](#)
 moving contents [17](#)

printer

 location codes [169](#)

pwd command [122](#)

Q

quorums

 changing to nonquorum status [9](#)
 definition [2](#)
 nonquorum volume groups [8](#)

R

Range setting [58](#)

raw logical volume

 define [48](#)

reading the three-digit display [224](#)

recovering data from a disk without reformatting [28](#)

recovery procedures for failed disk drive

 example of [32](#)

relative path name [120](#)

relocating

 adapter for DLPAR [238](#)

remote

 login [224](#)

removing

 local display [231](#)

renaming

 directories [121](#)

restore

 effect of fragments on [116](#)

restricting users from specified directories [28](#)

rmdir command [125](#)

root (/) file system [76](#)

root file system [74](#)

root volume group (rootvg)

 mirroring [46](#)
 reducing size of file systems [90](#)

S

SCSI devices

 location codes [170](#)

Shared Product Object Tree (SPOT) directory [81](#)

skulker command [28](#)

software

 checking for device problems [195](#)

space

 displaying available [83](#)

splitting a mirrored disk from a volume group [26](#)

SPOT directory [81](#)

starting Workload Manager [133](#)

stopping Workload Manager [133](#)

strict inter-disk setting [59](#)

su command [225](#)

super strict inter-disk setting [59](#)

swap space

 see paging space [65](#)

system

 accounting [224](#)

 displaying name [228](#)

 powering on [224](#)

T

tape drives

 attributes

 changeable [200](#), [202–210](#)

 managing [200](#)

 special files for [210](#)

targeted device configuration [199](#)

three-digit display [224](#)

touch command [226](#)

tty (teletypewriter)

 location codes [169](#)

U

uname command [227](#), [228](#)

unmirroring

 volume group [49](#)

USB Blu-ray drive support [213](#)

USB device support [212](#)

USB flash drive [212](#)

user

 changing to another [225](#)

user ID

 changing to another [225](#)

user-defined volume groups

 importing [90](#)

users

 displaying system ID [226](#)

 displaying who is logged in [228](#)

V

variable number of i-nodes

 and fragments [107](#)

vary-on process

 overriding failure of [36](#)

verifying file systems [89](#)

VGDA (volume group descriptor area) [1](#)

- VGSA (volume group status area) [1](#)
- Virtual Memory Manager [72](#)
- Virtual Memory Manager (VMM)
 - overview [65](#)
- VMM [72](#)
- volume group
 - mirroring [46](#)
 - root
 - mirroring [46](#)
 - splitting a mirrored disk from [26](#)
 - unmirroring [49](#)
- volume group descriptor area (VGDA) [1](#)
- volume group status area (VGSA) [1](#)
- volume groups
 - changing to nonquorum status [9](#)
 - definition of [42](#)
 - exporting [16](#)
 - high availability [54](#)
 - importing [16](#)
 - moving [16](#)
 - nonquorum [8](#)
 - policy implementation [64](#)
 - quorums [2](#)
 - replacing a disk [34](#)
 - strategy for [54](#)
 - user-defined
 - importing [90](#)
 - vary-on process [1](#)
 - when to create separate [54](#)

W

- who am i command [227](#)
- who command [227](#), [228](#)
- whoami command [227](#)
- WLM
 - API [163](#)
- working directory [120](#)
- Workload Manager
 - API [163](#)
 - starting and stopping [133](#)
- write scheduling policy [56](#)
- write-verify policy [62](#)

X

- X terminal [231](#)

Z

- zero file allocations [112](#)

