IRIX NetWorker™ DataBase Module for Informix® Administrator's Guide

Document Number 007-3556-001

CONTRIBUTORS

Written by Vickie Brown and Bill Tuthill
Production by Michael Dixon
Engineering contributions by Legato employees

IRIX NetWorker™ DataBase Module for Informix® Administrator's Guide
Document Number 007-3556-001

Contents

List of Figures

List of Tables

About This Guide

The *IRIX NetWorker Database Module for Informix Administrator's Guide* contains information on how to configure and manage the NetWorker Database Module for Informix database storage management software.

Use the information in this guide in conjunction with the *IRIX NetWorker Administrator's Guide* and the document set provided with your INFORMIX® – OnLine Dynamic Server software.

Intended Audience

This book is intended for system administrators and database administrators (DBAs) who install software and maintain OnLine Dynamic Server on a network.

The information presented here provides guidelines for using and administrating Database Module for Informix (DBMI) in a typical network environment. For details on using the NetWorker command line interface, refer to the online NetWorker reference pages after you install the software.

Software Requirements

To run DBMI, systems must have the following:

- IRIX NetWorker release 4.2.5 or later must be installed on the storage management server.
- NetWorker client software, release 4.2.5 or greater, must be installed on the system running OnLine Dynamic Server. If the NetWorker software is not installed, refer to the *IRIX NetWorker Installation Guide* for instructions.

Conventions

The following conventions are used in this manual to make information clear and accessible.

- Command and program names are displayed in **bold** typeface. For example:

  To start the NetWorker Administrator program, use the **nwadmin** command.

- Examples of what *you* type are shown in **`bold fixed-width`** typeface. For example:

  # **`nwadmin &`**

- Text that you substitute as a variable is displayed in *italic* typeface. For example:

  `%` **`man`** *nsrCommand*

- Examples, shell prompts, and information displayed on the screen are displayed in `fixed-width` typeface. For example:

  ```
  media waiting: recover waiting for 8mm 5GB tape volume name
  ```

- Names of NetWorker buttons, displays, menus, scrolling lists, and windows are displayed in Helvetica typeface. For example:

  Click the Cancel button to close the Help window.

- Directory pathnames, machine names, and new terms defined in the glossary are displayed in *italic* typeface. For example:

  When you use NetWorker to back up your */usr* files from the client machine named *venus*, you are saving them to *backup volumes* on the NetWorker server.

The following paragraph types indicate various kinds of information you need to use NetWorker productively.

Caution: Important pieces of information and cautionary notes that prevent you from making a mistake are marked “Caution.”

Note: Helpful information that you should probably know about is marked “Note.”

Tip: Tips or suggestions that you do not necessarily have to follow, but may give you hints as to how to set up NetWorker at your site, are marked “Tip.”

Shortcut: Step-by-step procedures that help you save time because they provide the minimum information you need to complete a task are marked “Shortcut.”

Documentation and Support

Both the *IRIX NetWorker Administrator's Guide* and *IRIX NetWorker User's Guide* are available online, as IRIS InSight manuals. To view the manuals, first install the subsystems *networker4.books.NetWorker_AG* and *networker4.books.NetWorker_UG*, then run the insight(1) command.

To print out this manual, Silicon Graphics recommends that you download the PDF or PostScript® file from http://www.sgi.com/Technology/TechPubs. Click Library Search and search for "DataBase Module" in book titles. IRIS InSight was never intended as a hardcopy publishing solution.

To learn how to use the NetWorker Backup and Recover windows for manual backups, see the *IRIX NetWorker User's Guide*. To get information about a specific release of IRIX NetWorker, see the *IRIX NetWorker Release Notes*. If you would like more technical information about the NetWorker commands, see the online reference pages after you have installed NetWorker.

Silicon Graphics offers a comprehensive product support and maintenance program for IRIS products. For information about using support services for this product, refer to the *Release Notes* that accompany it.

Chapter 1

Components and Installation

NetWorker Database Module for Informix (DBMI), used with Informix—OnLine Dynamic Server and IRIX NetWorker, provides reliable, high-performance data protection for local or distributed OnLine Dynamic Server databases. DBMI integrates backup and restore procedures for OnLine Dynamic Server databases into the network-wide data protection facilities that NetWorker provides.

The Importance of Backing Up Mission Critical Data

Although the reliability of computer equipment has improved greatly in recent years, hardware failures still occur, sometimes with catastrophic results.

In a client-server environment, data can be lost not only after hardware failures, but also because of user errors. Software bugs or procedural flaws and simple user error are common culprits, requiring database media restores. A viable backup strategy can help you recover from these potentially disastrous situations.

Many database administrators do not recognize the danger of failing to make regular backups of database objects and frequent backups of a *logical log*. If all logical logs are lost, a database can only be recovered to the time of its last full backup. Without backups or logical logs, the database cannot be recovered at all.

INFORMIX ON-Bar

ON-Bar is a utility, included with Informix—OnLine Dynamic Server, that provides online database backup and restore services for an OnLine Dynamic Server *dbobject*. ON-Bar provides

- online, concurrent backups and restores of *dbspace*, *blobspace*, and logical log files
- automated, continuous logical log backup or on-demand logical log backups
- an interface to popular storage management software

IRIX NetWorker

IRIX NetWorker is a high-capacity, easy-to-use network data storage management solution that provides data backup and recovery for heterogeneous networks of servers and clients. NetWorker simplifies storage management and reduces administrative burden by automating and centralizing your data storage operations. With NetWorker, you can

- perform automated “lights-out” backups during off-peak hours
- use centralized administration to configure, monitor, and control backups from anywhere on a network
- automate tape handling tasks using SmartMedia™, an intelligent media manager that supports a wide variety of 4 mm DAT, 8 mm, and high-end devices, as well as bar code label recognition and cleaning cartridge support
- increase backup performance by simultaneously sending data from multiple clients to the backup server
- use concurrent device support to direct data streams to multiple backup devices for even greater speed

NetWorker Database Module for Informix

NetWorker DBMI is an add-on module for NetWorker that provides automated backup media management and scheduling for ON-Bar. DBMI provides

- true “lights out” database storage management through automated scheduling, *autochanger* support, and electronic tape labeling and tracking
- support for local backup or distributed backup to a centralized backup server
- high performance support for concurrent devices, such as DLT drives

Chapter 2 provides a detailed breakdown of the NetWorker *daemon* processes and a description of how they function during a scheduled backup and on-demand recover.

Installing the Components

There are three components to install: the NetWorker server, the database module, and Informix ON-Bar.

Installing the NetWorker Server

You must install a NetWorker server, version 4.2.5 or higher, if one is not already installed. In general, there are three steps:

1. Load the IRIX NetWorker distribution CD-ROM into a drive.
2. Run **inst** and install the appropriate product images.
3. Start the NetWorker daemons, license the server and jukebox (autochanger) support if applicable, and configure client backups.

Refer to the *IRIX NetWorker Installation Guide* for complete instructions on how to install, license, and configure the NetWorker server.

Installation Requirements for DBMI

Before installing the DBMI, you should have the following:

- a Silicon Graphics server running Informix and IRIX 6.2 or higher
- 5 MB of free disk space to store DBMI and provide space for new indexes
- a tape drive or jukebox that is compatible with IRIX NetWorker

Installing the NetWorker Database Module

Follow this procedure to install the NetWorker Database Module for Informix:

1. Load the IRIX NetWorker software distribution CD-ROM.
2. On the NetWorker server, switch user to root.

   ```
   % /bin/su -
   Password:
   ```

   If the NetWorker server is already running, stop it now:

   ```
   # /etc/init.d/networker stop
   ```

3. Run the **inst** command, specifying the location of the NetWorker distribution:

   ```
   # inst -f /CDROM/dist
   inst> list
   ```

4. Select subsystems inside **inst**. Use Table 1-1 as a guide to selecting product images. To install the DBMI software and documentation, enter these commands:

```
Inst> keep *
Inst> install networker4.sw.DBMI networker4.books.NetWorkerDBMI
```

For more information on the available product images, refer to the *IRIX NetWorker Release Notes*. For details about **inst**, refer to *IRIX Admin: Software Installation and Licensing*, or see the inst(1M) reference page.

Table 1-1 Subsystems for NetWorker Servers

Subsystem	Description	Required/Optional
networker4.sw.DBMI	Database Module for Informix, connector for ON-Bar	Optional; license required
networker4.books.NetWorkerDBMI	Online version of DBMI book	Optional; InSight needed

5. Perform the installation and exit **inst**:

```
Inst> go
...
Inst> quit
```

6. Configure NetWorker pools and label templates for DBMI with this command:

```
# /etc/dbmi_config
```

7. Start the NetWorker daemons with this command:

```
# /etc/init.d/networker start
```

You must purchase a NetWorker Database Module for Informix for every NetWorker server (machine) on which you want to use DBMI. Each NetWorker Database Module must be licensed separately. Refer to the section “Licensing NetWorker Severs” in the *IRIX NetWorker Installation Guide* for more information.

Installing INFORMIX ON-Bar

For specific ON-Bar installation instructions, refer to your *Informix—OnLine Dynamic Server Installation Guide*. If you encounter problems during this phase, please refer to the documentation, or contact Informix technical support.

Chapter 2

NetWorker Functionality

NetWorker client-server technology uses a network Remote Procedure Call (RPC) protocol to back up data. The NetWorker server software consists of several daemons and programs that oversee backing up and recovering, as well as storage management client configurations, an online client index, and an online media database. NetWorker client software also includes a client-side daemon and client-side programs.

This chapter describes how NetWorker functions. It provides a brief, simplified overview of how NetWorker performs a backup and recover. Illustrations of backup and recovery provide a graphical overview of the storage management that NetWorker provides. The storage management process employed with DBMI is also illustrated.

This information may be useful for system administrators responsible for storage management and protection of the data stored in an *RDBMS* on a network.

How NetWorker Backs up Data

NetWorker calls on several daemons and programs when it receives a request for backup. The daemons coordinate the tasks associated with a backup or recover and record information about what files were backed up and the media containing the backed-up data.

NetWorker Daemons and Programs

Table 2-1 provides a description of the NetWorker server daemons and programs, which contact the client for a backup and maintain the server's client index and media databases. The online NetWorker reference pages contain further details about the NetWorker daemons and programs.

Table 2-1 NetWorker Server Daemons

Daemon/Program	Function
ansrd	Monitors an active **save** or **recover** session; agent process called by **nsrd** in response to a **save** or **recover** session
asavegrp	Monitors progress of individual save sets; agent process called by **savegrp**
nsrck	Checks the consistency of the online file index; invoked by **nsrd** whenever the consistency of the online file index needs to be confirmed
nsrd	Provides an RPC-based **save** and **recover** service to NetWorker clients; master NetWorker daemon
nsrim	Automatically manages the server's online client index; invoked at the end of a **savegrp**
nsrindexd	Provides a method for inserting entries in the online client index based on information passed by **save**
nsrmmd	Provides device support, to generate mount requests, and *multiplex* save set data during a multi-client backup—**nsrd** can start several **nsrmmd** daemons, up to the number specified in the NetWorker server's "active devices" attribute; media multiplexor daemon
nsrmmdbd	Provides media and save set database management services to the local **nsrd** and **nsrmmd** daemons and records entries in the media database; media management database daemon
savegrp	Runs a group of NetWorker clients through the **save** process

The master NetWorker server daemon, **nsrd**, is responsible for several tasks:

- starting the other server daemons
- allocating media daemons
- authorizing backup and recover services for the client
- contacting clients for scheduled backups
- maintaining NetWorker configuration information
- monitoring backup and recover sessions
- maintaining server statistics and message logs

The NetWorker server daemons call on the NetWorker client daemon, **nsrexecd**, and several client-side programs when a scheduled or on-demand backup request is received. A temporary server agent daemon, **ansrd**, is started to allow the NetWorker server to monitor the progress of the backup session. Table 2-2 provides a description of the client-side daemons and programs.

Table 2-2 NetWorker Client Daemons

Daemon/Program	Function
nsrexecd	Authenticates the NetWorker server's remote execution request and executes the **save** and **savefs** commands on the client
recover	Browses the NetWorker server's online client index and restores the specified file to primary disk storage
save	Sends specified files in a multiplexed data stream to the NetWorker server for entry in the online indexes by **nsrindexd** and eventual backup to media by **nsrmmd**
savefs	Sends information about the save sets to back up for the client; identifies save set data modified since the last level **save**

What Happens During a Scheduled NetWorker Backup?

When you configure a *backup group* on the NetWorker server, you schedule a start time for the backup group. The **nsrd** server daemon starts the server's **savegrp** program for the backup group at the scheduled time.

The **savegrp** program queries the NetWorker server to determine

- which clients configured on the server are members of the scheduled group
- what level of **save** to perform
- how many save sets to run concurrently, determined by the parallelism value set on the NetWorker server
- when the last backup of the group occurred

If any of this information is not available on the NetWorker server, **savegrp** sends a request to the client-side daemon **nsrexecd** to run **savefs** on each client assigned to the backup group to gather the necessary details.

The **savefs** program is responsible for letting **savegrp** know which objects are supposed to be backed up for the client. Once **savegrp** receives information about the objects to back up, **savegrp** assembles a work list for the server.

If problems were encountered with the online index during the last backup session, **nsrd** invokes the **nsrck** daemon to check the consistency and state of the NetWorker server's online indexes. Then, **nsrd** starts the online file index insertion daemon, **nsrindexd**.

The **savegrp** program contacts the first client on the server's work list. The client's **nsrexecd** daemon is invoked and starts a save session of the first save set listed on the server's work list. The **save** program passes to **nsrd** all save criteria, such as group, client, save sets, and level of the save data. With this information, **nsrd** determines the pool of volumes that store the data and forwards the information to the media daemon.

The media daemon, **nsrmmd**

- sends a message to the console of the NetWorker server, requesting a mount of the media assigned to the volume pool indicated by **nsrd**
- writes the data sent by **save** to storage media
- forwards storage information to **nsrmmdbd** for recording in the NetWorker server's online media database

Anytime there is a lull in save set activity from the client, the NetWorker server attempts to find another save set in the group to keep the process moving along. The **savegrp** program attempts to concurrently back up as many save sets as possible, up to the limit set by the parallelism attribute in the NetWorker server's configuration, so as to use the backup devices to their maximum potential.

The **savegrp** program repeats the process for each item on the server's work list until all clients in the group are backed up. Before the **savegrp** completes, **nsrim** is invoked and the NetWorker server's *bootstrap* save set is backed up. The final results of the **savegrp** execution are sent back to the server and are included in the "savegroup completion" notification that NetWorker e-mails to *root*. (Refer to Chapter 7, "Customizing NetWorker Backups," in the *IRIX NetWorker Administrator's Guide* for details on using and customizing NetWorker notifications.)

Note: For DBMI, the "savegroup completion" notification is produced for dbobject backups only— for information on logical log backups, use the **nsrinfo** command, or query the **bar_action** and **bar_object** tables for information on the backup and restore activities for OnLine Dynamic Server dbobjects.

Refer to the *IRIX NetWorker Administrator's Guide* or the nsrinfo(1M) reference page for help on using **nsrinfo**. Refer to Chapter 4, "Catalog Tables," in the *INFORMIX-Online Dynamic Server Backup and Restore Guide* provided with your OnLine Dynamic Server software for information about ON-Bar catalog tables.

Figure 2-1 shows a flowchart of NetWorker client and server daemons and programs during a scheduled save.

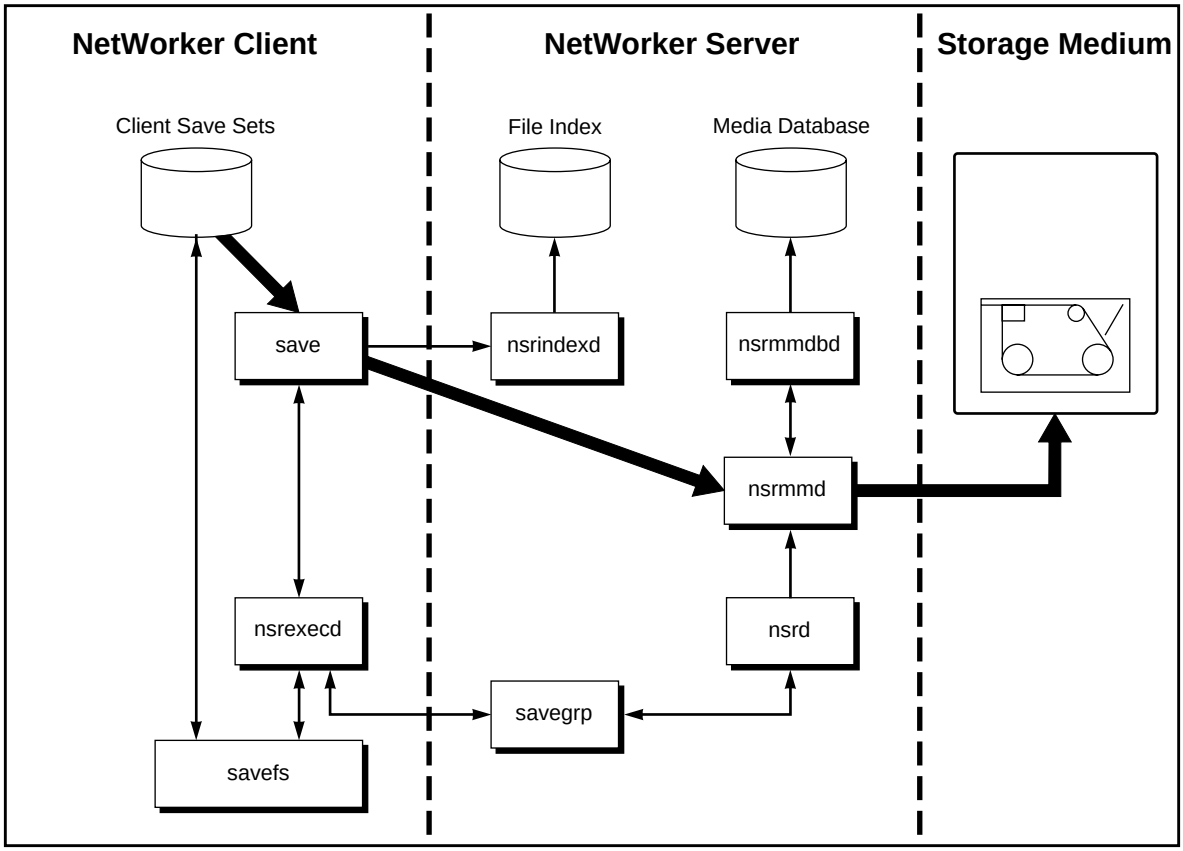

Figure 2-1 NetWorker Daemons During Scheduled Save

This is how NetWorker daemon processes and programs interact during a scheduled save. Heavy lines represent movement of data from local disk to storage medium.

How NetWorker Recovers Data

When NetWorker receives a recover request from a client, the server's **nsrd** daemon contacts the server's media daemon, **nsrmmd**. This daemon contacts the server's media database daemon, **nsrmmdbd**, to determine which media contain the save set requested by the **recover** command. Once it obtains the save set's media location, **nsrmmd** issues a mount request, the media is positioned at the beginning of the save set, and the save set stored on the mounted media is passed to **nsrmmd**. The media daemon forwards the save set to the client's **recover** program, which restores data to the client's filesystem.

Figure 2-2 shows a flowchart of NetWorker server and client daemons and programs during recovery of NetWorker client data.

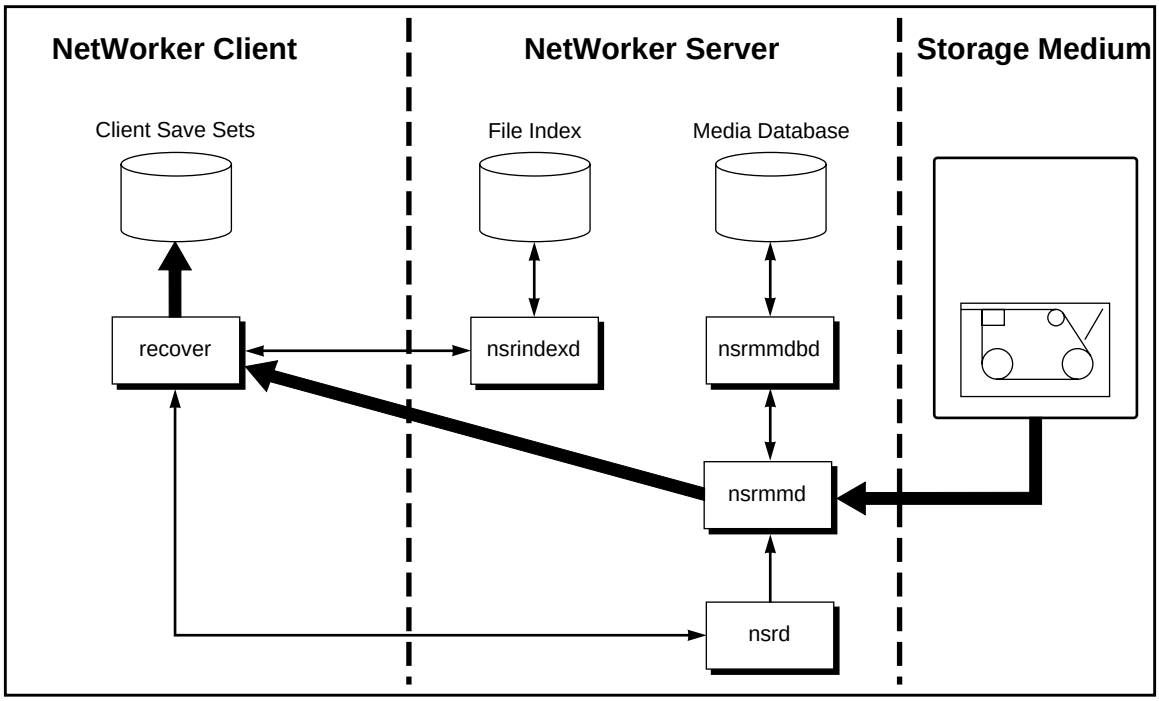

Figure 2-2 NetWorker Daemons During Recover Session

This is how NetWorker daemon processes and programs interact during a recover session. Heavy lines represent movement of data from storage medium to local disk.

How DBMI Connects NetWorker to ON-Bar

NetWorker and DBMI provide storage management services for OnLine Dynamic Server dbobjects through an X-Open® backup services (XBSA) connection to ON-Bar. DBMI provides the services that connect NetWorker functionality to ON-Bar. You use the NetWorker administration program to set up backup schedules, label backup volumes, and configure the system running OnLine Dynamic Server as a storage management client of the NetWorker server.

When **nsrd** triggers a scheduled backup for an OnLine Dynamic Server instance on the NetWorker server, **savegrp** executes the **nsrdbmi** script instead of performing a standard save. The **nsrdbmi** script invokes ON-Bar, which interacts with NetWorker through the XBSA API to coordinate a backup of the specified OnLine Dynamic Server dbobjects.

Once the scheduled backup completes, ON-Bar performs a backup of the logical logs associated with the saved ON-Bar dbobjects, closes and backs up the current logical log, and opens a new log. Then NetWorker performs a "full" backup of the OnLine Dynamic Server instance's emergency boot file and server configuration file. Finally, NetWorker copies the server's bootstrap save set to tape, and e-mails a "savegroup completion" notification.

NetWorker takes care of the scheduling and storage management tasks, while ON-Bar takes care of passing the data from OnLine Dynamic Server to NetWorker.

Figure 2-3 shows the functional relationship between NetWorker, DBMI, ON-Bar, and OnLine Dynamic Server during backup. NetWorker connects to ON-Bar through an XBSA API to provide scheduled backups. Heavy lines represent movement of data from local disk to storage medium.

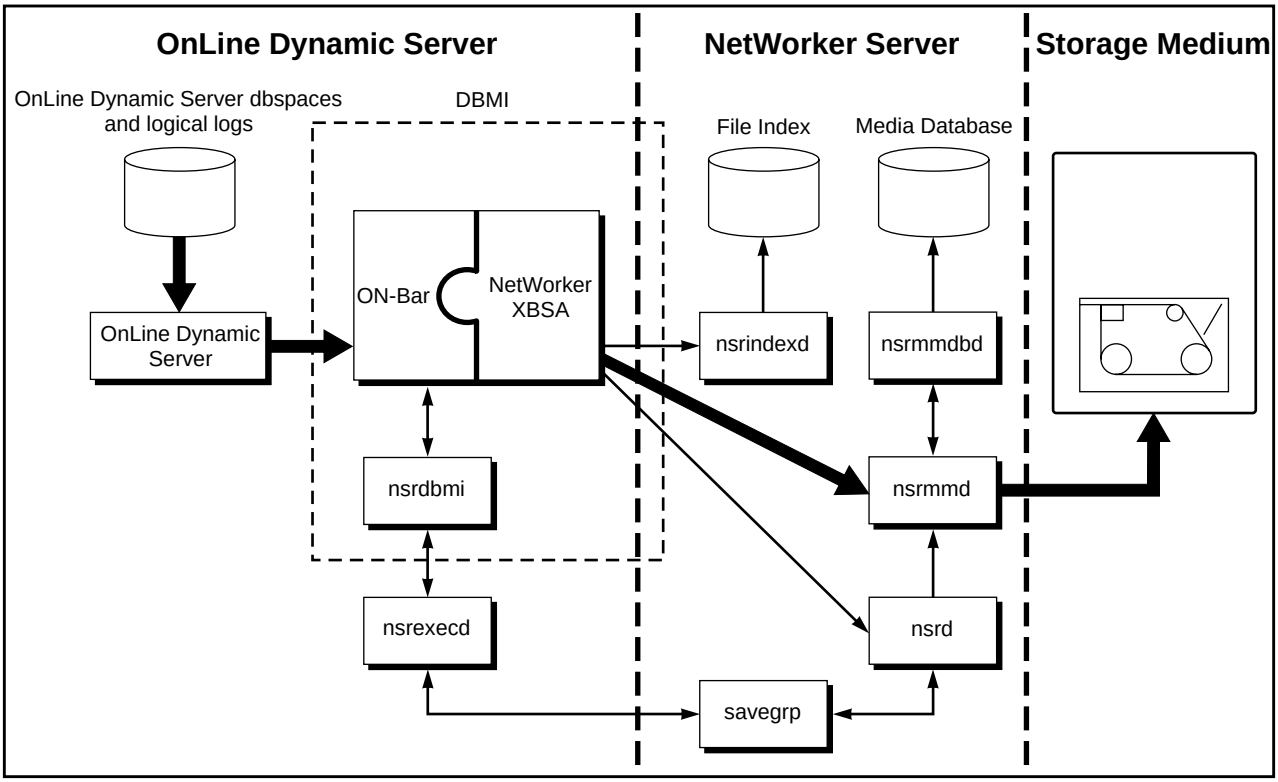

Figure 2-3 NetWorker Connection to ON-Bar

The OnLine Dynamic Server software can exist on the same system as the NetWorker server software, or it can exist on a separate system. Since **nsrdbmi** substitutes for the client-side program **save** during a backup, the DBMI software must be installed on the system where you installed OnLine Dynamic Server. No matter where the system running OnLine Dynamic Server resides, the system is considered a storage management client of the NetWorker server.

When an ON-Bar restore request is initiated, the NetWorker XBSA API translates the object names requested by ON-Bar into a format understood by NetWorker and forwards it to the NetWorker server's **nsrd** daemon. The media daemon, **nsrmmd**, searches the NetWorker server's online media database for the media containing the requested objects and recovers the data to the OnLine Dynamic Server as described in "How NetWorker Recovers Data" on page 10.

Figure 2-4 shows the functional relationship between NetWorker, DBMI, ON-Bar, and OnLine Dynamic Server during restore.

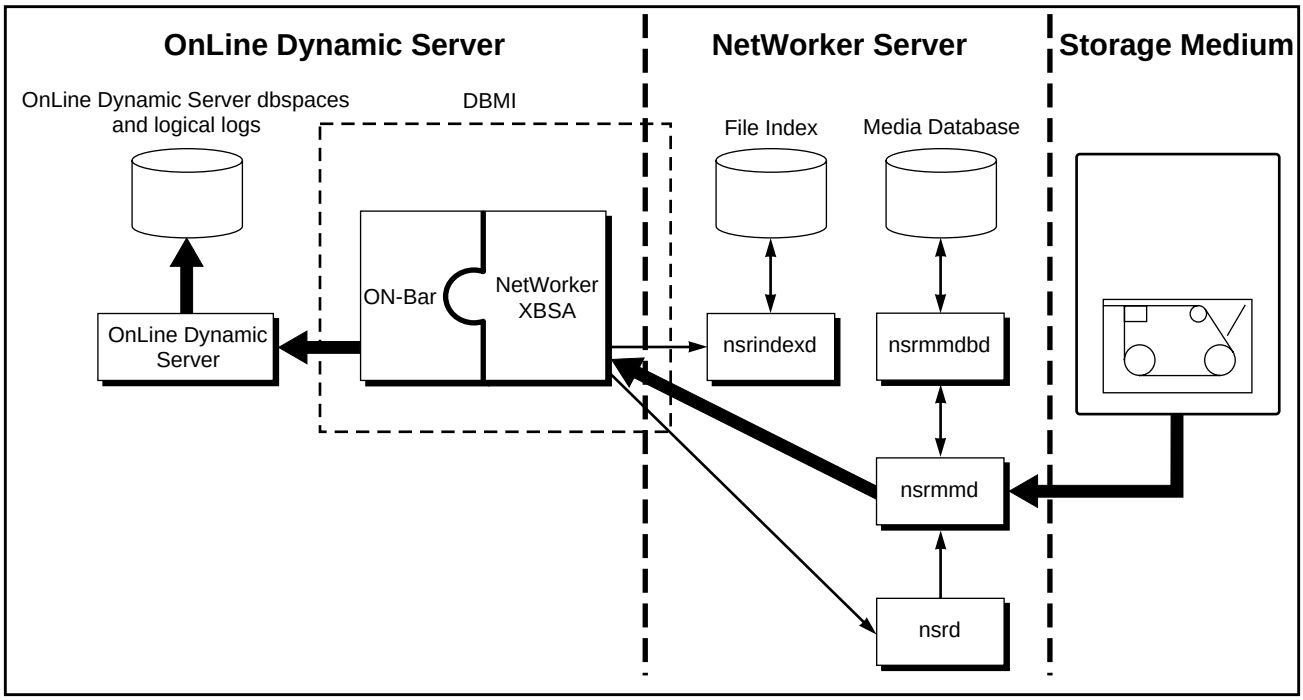

Figure 2-4 Restore Initiated by ON-Bar

This shows data movement during a restore initiated by ON-Bar. Heavy lines represent movement of data from storage medium to local disk.

Chapter 3 provides instructions for configuring OnLine Dynamic Server as a NetWorker client for scheduled backups.

To view the NetWorker server's index entries for dbobjects backed up for an OnLine Dynamic Server instance, use the **nsrinfo** command:

```
# nsrinfo -s jupiter -n informix -X informix mars
scanning client `mars' for all savetimes
/venus/rootdbs/0, rootdbs, 1.2 MB, Mon Aug 26 12:05:44 1996, full
/venus/01/29, logical log, 123 KB, Mon Aug 26 12:02:39 1996,
3 objects found
```

Refer to the nsrinfo(1M) reference page and the *IRIX NetWorker Administrator's Guide* for complete information on using the **nsrinfo** command.

Alternatively, you can query the ON-Bar catalog tables **bar_action** and **bar_object** to determine which backup or restore action occurred on database objects in an OnLine Dynamic Server instance. Refer to the documentation included with the OnLine Dynamic Server software for details on querying catalog tables for information.

The *IRIX NetWorker Administrator's Guide* provides complete details about the reports generated by NetWorker. For suggestions on using these reports as a part of your disaster recovery plan, refer to the documentation provided with your NetWorker server software.

Chapter 4 provides information on performing a manual, on-demand backup of OnLine Dynamic Server.

Viewing the Results of a Backup

NetWorker provides several reports about the results of a backup:

- An e-mail “savegroup completion” notification upon completion of a scheduled backup. You can edit the notification setup for this report, using the Customize manu of the **nwadmin** program. See “Setting up Event Notification” in the *IRIX NetWorker Administrator’s Guide* for details about customizing notifications.
- A series of messages written to the NetWorker message log files. Refer to Appendix B, “Error Messages.” for more information about NetWorker and NetWorker XBSA messages.
- A scrolling list of messages displayed in the main window of the NetWorker administrative graphical interface:

  ```
  Mon 11:56:00 media event cleared: backup to pool ‘DBMIData’ waiting
  for 1 writable backup disk or tape
  Mon 11:56:39 mars:INFORMIX:/venus/dbspace01 saving to pool
  ‘DBMIData’ (DBMIData.001)
  Mon 11:59:15 mars:INFORMIX:/venus/dbspace01 done saving to pool
  ‘DBMIData’ (DBMIData.001)
  ```

- A scrolling list of messages displayed in the Group Control window of the NetWorker administrative graphical interface. These messages are displayed in three lists: pending save sets, completed save sets, and failed save sets. You have the option of printing a columnar version of the details displayed in the Group Control window—first choose Tabular from the View menu.
- A printout of the NetWorker server’s bootstrap information for the backup session, showing the date, time, level, save set ID, file position in the save set entry, and the volume(s) to which the save sets were written (the last entry is for the backup of the server’s bootstrap save set):

  ```
  August 26 01:30 1996 mars’s bootstrap information Page 1

  date     time      level   ssid    file   record  volume

  8/26/96  11:59:15  full    16540   1      0       venus.DBMIData.001
  8/26/96  12:02:39  full    16564   1      0       venus.DBMILogs.001
  8/26/96  12:05:44  full    16566   1      0       venus.001
  ```

The completion reports do not show information distinguishing individual dbobject names.

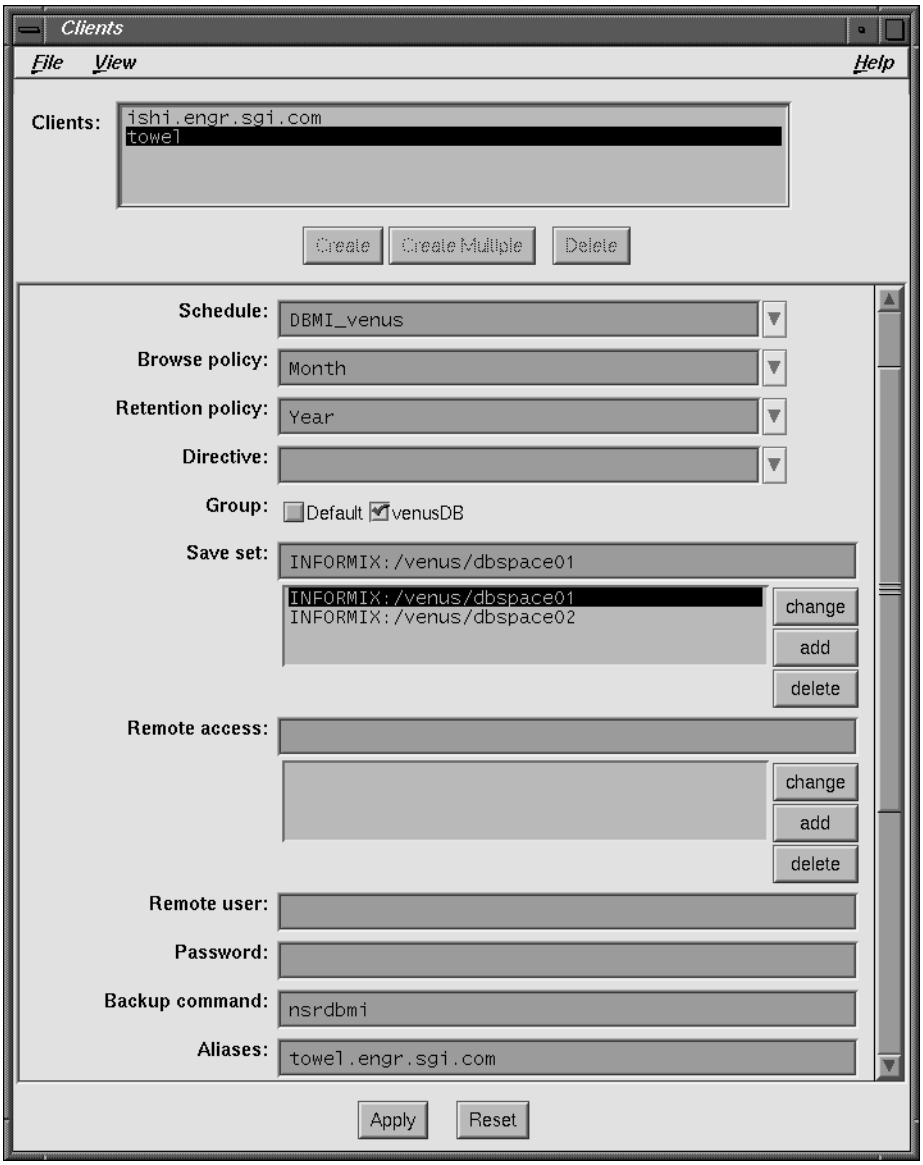

Figure 3-4 Database Server as the DBMI Client

5. Specify a *save set*. If you want to schedule an instance backup, enter the instance name. The entry shown in the example is the equivalent of performing an **onbar -b -L** *level* on the "venus" instance:

   ```
   INFORMIX:/venus
   ```

   If you want to schedule a selected-dbobject backup, include the dbspace or blobspace name in the save set string. You can specify more than one dbobject by making a separate save set entry for each dbobject. The entry shown in the example is the equivalent of performing **onbar -b -L** *level* **dbspace01** on the "venus" instance:

   ```
   INFORMIX:/venus/dbspace01
   ```

   Caution: To perform a whole-system backup, you must use the ON-Bar command line interface. See Chapter 4 for information on invoking on-demand DBMI backups using the ON-Bar command-line interface.

   Refer to the *INFORMIX—Online Dynamic Server Backup and Restore Guide* included with your OnLine Dynamic Server software for further details about the types of backups supported by ON-Bar.

   Tip: You can set up multiple instances of OnLine Dynamic Server that exist on the same system as separate NetWorker client resources. Create a custom copy of the **nsrdbmi** script and change the value assigned to the ONCONFIG variable to reflect the value assigned for the instance.

6. Enter **nsrdbmi** or the pathname of a custom **nsrdbmi** in the Backup command field.
7. Specify all known aliases for the system where OnLine Dynamic Server is installed.
8. Leave the attributes for directives, remote access, remote user, and archive users blank. Data compression is controlled by the **nsrdbmi** script.
9. Click Apply to save the client resource.

   Tip: To send an e-mail notification of the results of a scheduled backup to the owner of a save set, use the "view details" option (for UNIX) or "expert mode" (for NT) to edit the NetWorker client resource for the OnLine Dynamic Server instance. Scroll down to the Owner notification field and enter a notification command directed to the login ID of the owner. For example:

   ```
   /usr/ucb/mail -s "jupiter-mars-venus backup" jdoe
   ```

Refer to Chapter 3, "Configuring and Monitoring Clients," in the *IRIX NetWorker Administrator's Guide* shipped with your NetWorker server software for more details on using the NetWorker administrative graphical interface to configure a client.

You should keep copies of your logical log file backups until the associated dbobject save sets have exceeded their browse policy.

Caution: NetWorker does not allow a browse policy to exceed its retention policy. An entry for a save set must be removed from the file index before the save set can be removed or marked recyclable in the media index.

NetWorker Backup Clients

NetWorker uses a client-server model to provide storage management services. At least one machine on the network is designated as the NetWorker server. Machines with data to back up are configured as clients of the NetWorker server.

You can configure NetWorker clients using the Clients window of the NetWorker administration program. NetWorker maintains the resource information and contacts clients listed in a backup group configured on the server, performs on-demand backups when a client request is received, and restores data upon request from the client.

The NetWorker server maintains the online file and media indexes.

What is a NetWorker Client?

A NetWorker client is a resource configured on the NetWorker server. The client resource provides the server with information about the data to back up for a client, how long to maintain entries for the data in the online index for recovery (browse policy), and how long to keep the media containing the client's backed-up data (retention policy).

Creating a NetWorker Client

To set up a system running OnLine Dynamic Server as a NetWorker client:

1. Run the **nwadmin** program on the NetWorker server.
2. Choose Client Setup from the Clients menu; the Clients window appears, as shown in Figure 3-4.
3. Select a browse policy and a retention policy.
4. Select a group. You might want to create a custom group for database backups.

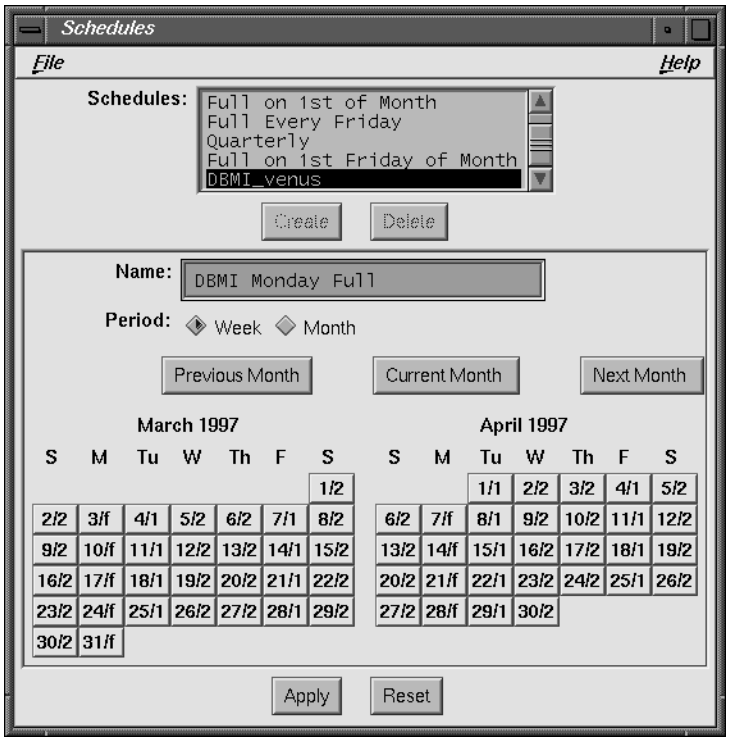

Figure 3-3 Custom DBMI Schedule

Caution: If you use a value other than those listed for NetWorker in Table 3-1, backups will fail and generate the error message “Only level 0 (full), 1, or 2 backups allowed.”

Using NetWorker Policies

NetWorker uses browse policies for the client index entries and retention policies for the media index entries to manage and reduce the size of the online indexes. You can choose one of the preconfigured policies provided with NetWorker or create policies of your own. A policy can be used as either a browse or a retention policy.

You can also manage indexes manually using the Indexes and Volumes windows. See “Manually Managing the Online Indexes” in Chapter 3 of the *IRIX NetWorker Administrator’s Guide* for a discussion of manual index management and index policy.

NetWorker provides several preconfigured schedules for use in backing up filesystems. These schedules support backup levels not available with ON-Bar. You need to create a custom schedule for backups of your OnLine Dynamic Server.

The **nsrdbmi** script translates ON-Bar backup levels to valid NetWorker levels, as shown in Table 3-1. Create a schedule for your OnLine Dynamic Server backups, using the information provided in Table 3-1 and the instructions provided in "Setting up Backup Schedules" in the *IRIX NetWorker Administrator's Guide*.

If your OnLine Dynamic Server manages a significant amount of data, consider scheduling a full backup of an instance every one to two weeks and incremental backups on the other days to protect data that changes between full backups.

Table 3-1 ON-Bar and NetWorker Backup Levels

ON-Bar	NetWorker	Amount of Data Backed Up
0	full	All pages containing data for the instance listed in the save set entry
1	1	Pages that have changed since the last level full backup
2	2	Pages that have changed since the last level 1 backup
	skip	Skip the scheduled backup—no data is backed up

Figure 3-3 shows a custom backup schedule that performs a full backup every Monday, followed by level 1 backups on Tuesday and Friday. The remaining days are scheduled as level 2 backups.

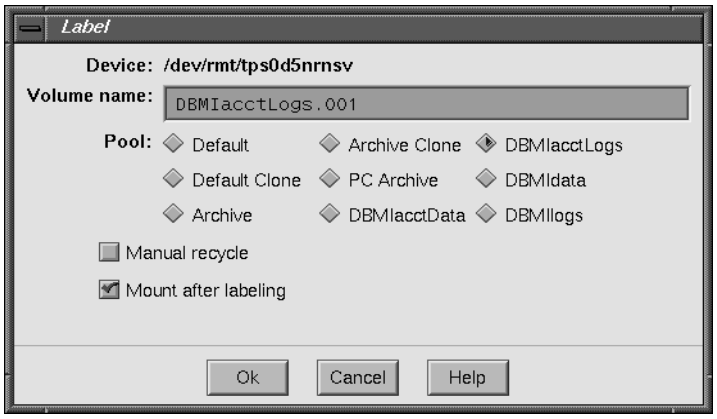

Figure 3-2 Label Window on NetWorker Server With Custom Pool

Modifying the Pool Variables in the nsrdbmi Script

Once you have added custom data and logical log volume pools to the NetWorker server's resource database, you need to copy and modify the **nsrdbmi** script to assign the new values to the NetWorker XBSA pool variables:

- NSR_DATA_VOLUME_POOL
- NSR_LOG_VOLUME_POOL

Refer to "Using Volume Pools" in the *IRIX NetWorker Administrator's Guide* shipped with your NetWorker server software for complete instructions on creating NetWorker pools.

Using NetWorker Backup Schedules

NetWorker uses backup schedules to automate the level of a scheduled backup. The NetWorker server's administration program provides a calendar that accepts entries designating the level of backup that should occur on a given day of the week.

Tip: You can start a scheduled backup at any time by using the Group Control window in the NetWorker administration program. Refer to "Monitoring and Controlling Backups" in the *IRIX NetWorker Administrator's Guide* for instructions on using this feature.

Creating a Volume Pool for DBMI

By default, NetWorker and DBMI use the DBMIData volume pool for dbspaces and blobspaces, and the DBMILogs volume pool for logical logs associated with backed-up dbobjects.

To create a custom pool for DBMI, choose Pools from the Media menu of **nwadmin**, and in the Pools window:

1. Create a name for the data pool.
2. Select Backup for the pool type.
3. Select a custom label template you created.
4. Enter the pool's name in the Save sets field and add it to the list. DBMI only uses this designation to meet the NetWorker requirement for having at least one attribute defined for a pool.
5. Select a device from the choices displayed.
6. Apply the settings.
7. Repeat steps 1 through 6 for the logical logs pool.

When you click the Apply button, NetWorker adds the pool to its list of volume pools. The pool is also added to the choices available for labeling volumes. When you select a pool for labeling volumes, NetWorker uses the label template you created and assigned to the pool to display the next label available in the series, as shown in Figure 3-2.

Note: The label template's name cannot include any of the characters reserved for use as field separators. NetWorker displays a "Character invalid in name…" message if you try to save a configuration that contains a field separator in the label template name.

After you create a label template and click the Apply button, NetWorker displays the label it applies to the next volume in the pool associated with the label. The label template is added to the list of label template choices available for NetWorker volume pools.

Figure 3-1 provides an example of a customized label template.

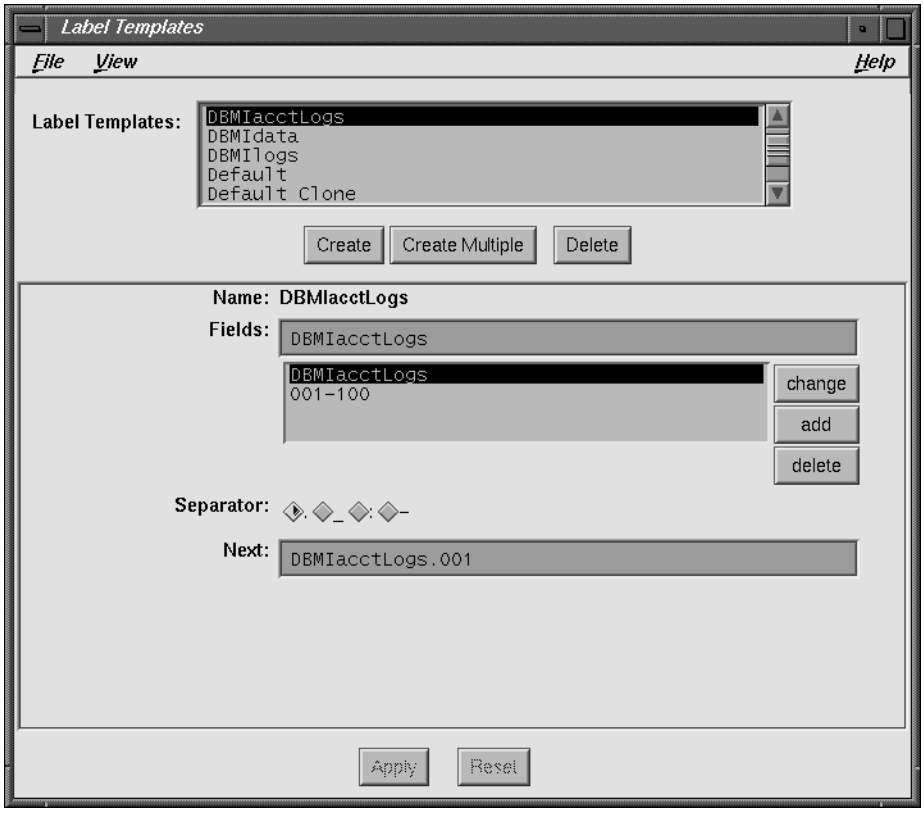

Figure 3-1 Custom Label Template on NetWorker Server

Refer to "Customizing NetWorker Backups" and "Labeling and Mounting Backup Volumes" in the *IRIX NetWorker Administrator's Guide* shipped with your NetWorker server software for complete instructions on using NetWorker label templates.

DBMI uses the NetWorker XBSA environment variables NSR_DATA_VOLUME_POOL and NSR_LOG_VOLUME_POOL to determine where to send OnLine Dynamic Server backup data—you do not need to assign a group or client to the pool. The **nsrdbmi** entry in the client Backup command relays the information needed to direct dbspace, blobspace, and logical log backups to the correct backup media.

Caution: If you elect to have ON-Bar back up logical logs automatically as they become full, modify the automatic log backup script *$INFORMIXDIR/etc/log_full.sh* on the system running OnLine Dynamic Server to include the lines:

```
NSR_LOG_VOLUME_POOL="DBMILogs"
NSR_SERVER=NetWorker_servername
export NSR_LOG_VOLUME_POOL
export NSR_SERVER
```

If you customize a pool for log file backups, replace DBMILogs with the name of the custom pool.

Continuous log file backups require a dedicated device and media availability. See "Performing Continuous Logical Log Backups" in Chapter 4 for more information on continuous log file backups.

You may want to organize backup data even further, for example, by department, type of database maintained, or level of backup. To customize media organization for DBMI do the following:

- Create label templates for OnLine Dynamic Server data and logical log backups.
- Create a volume pool.
- Copy the **nsrdbmi** script template, and modify values assigned to the NetWorker XBSA environment variables NSR_DATA_VOLUME_POOL and NSR_LOG_VOLUME_POOL.

Creating a Volume Label Template

NetWorker generates labels for backup volumes according to the rules of a label template configured on the NetWorker server. To create a new label template, you devise a name for the label template, specify the fields to use in the label, provide an alphabetic or numeric range for the volumes, and select a separator to use between fields. The order in which you enter the fields determines the order of the fields in the label template. The first field you enter is the first field NetWorker uses in the label template.

What Is a Volume Pool?

A volume pool is an assigned collection of backup volumes containing specific data sorted during a NetWorker backup. All NetWorker volumes belong to a pool, either one preconfigured by NetWorker or one you create.

In the **nwadmin** Pools window, each pool has a Pool type designation, indicating whether the volume contains data that has been archived, backed up, or cloned. For DBMI, the only valid pool types are Backup and Backup clone.

Volume pools provide the ability to segregate backed-up data, such as Informix dbspaces, blobspaces, and logical logs. Pools also allow you to direct backup data to specific devices.

You choose how NetWorker sorts backups, sending data to specific volumes labeled for the pool. You can sort NetWorker filesystem backup data by several categories:

- backup group
- backup level
- NetWorker client
- object type
- save set

For more information about using NetWorker volume pools see Chapter 5, “Managing Media and Backup Devices,” in the *IRIX NetWorker Administrator’s Guide*.

Customizing Volume Pools

As part of the DBMI installation process, you ran a script named **dbmi_config**, which created two volume pools and their associated label templates on the NetWorker server:

- the DBMIData pool and a label template for dbspace and blobspace backups
- the DBMILogs pool and a label template for logical log backups

The pool names DBMIData and DBMILogs are assigned to their respective NetWorker XBSA environment variables, `NSR_DATA_VOLUME_POOL` and `NSR_LOG_VOLUME_POOL`, in the **nsrdbmi** client script on the system running OnLine Dynamic Server.

- Client retries = 0
- Clones = No
- Clone pool = Default Clone

  Note: You can modify the Default group's attributes, but you cannot delete Default from the list of NetWorker groups.

Suggestions for Setting Up DBMI Groups

If you have a large number of OnLine Dynamic Server instances, consider creating backup groups with different start times to help reduce network traffic. You can have any number of backup groups configured on your NetWorker server.

When you select a start time for each group, be sure to schedule them far enough apart so one group completes its backup before the next group begins.

To send a copy of the server's bootstrap report to a specific printer, edit the group's resource using the "view details" option (for UNIX), or "expert mode" (for NT). Enter the designated name of the printer in the printer attribute for the group. Refer to the *IRIX NetWorker Administrator's Guide* and the nwadmin(1M) reference page for details on using the "view details" and "expert mode" options.

For instructions on how to set up a backup group, refer to the section, "Configuring Backup Groups," in the *IRIX NetWorker Administrator's Guide*.

Caution: Be sure to enable the Autostart option for the group you configure. Otherwise, the group's scheduled backup will never occur.

Using Volume Pools

NetWorker provides a means for directing your backups to specific sets of media. Volume pools allow you to establish a logical and systematic method for tracking, organizing, and sorting your backup data. Volume pools always have a label template associated with them, so as to provide an automated method of identifying media assigned to a pool.

3. Add the NetWorker XBSA environment variable to the list of variables exported.

```
export_environment_variables()
{
    export NSR_VOLUME_POOL
    ...
    export new_NSR_variable
}
```

For complete details on the preconfigured settings and valid values for NetWorker XBSA environment variables, see Appendix A, “XBSA Environment Variables.”

Caution: NetWorker and ON-Bar rely on shared XBSA libraries for interaction during backups and restores. Create a symbolic link, using the appropriate shared library file extension for your system. In the following example, which creates a symbolic link between the NetWorker and ON-Bar XBSA libraries, “so” indicates the file extension for most UNIX systems.

```
# ln -s /usr/lib/libbsa.so /usr/lib/ibsad001.so
```

Using a Backup Group

You can assign one or more OnLine Dynamic Server instances to a group of NetWorker clients. NetWorker groups allow you to distribute backups to alleviate network traffic and schedule backups for a time of day when performance demands on your database and NetWorker servers are lower.

What Is a Backup Group?

A backup group is a NetWorker server *resource* that allows NetWorker clients assigned to a group to start backing up files at a designated time once you enable the autostart feature. Data backed up for a NetWorker group can be written to a specific set of media, or allowed to mix with backup data from other NetWorker groups.

NetWorker provides a preconfigured group named “Default,” which is shipped with these preconfigured settings:

- Autostart = Disabled
- Start time = 3:33

Changing the Value for ONCONFIG

The ONCONFIG environment variable defines which OnLine Dynamic Server configuration file to use. The ONCONFIG variable in the **nsrdbmi** script is already assigned the value:

```
ONCONFIG=onconfig.std
```

If your OnLine Dynamic Server instance uses a different configuration filename, change the value assigned to ONCONFIG in the **nsrdbmi** script.

Changing the Value for INFORMIXSQLHOSTS

The INFORMIXSQLHOSTS environment variable defines where the *sqlhosts* file used by OnLine Dynamic Server resides. The INFORMIXSQLHOSTS variable in the **nsrdbmi** script is already assigned the value:

```
INFORMIXSQLHOSTS=$INFORMIXDIR/etc/sqlhosts
```

If the *sqlhosts* file used by OnLine Dynamic Server is in a different path, change the value assigned to INFORMIXSQLHOSTS in the **nsrdbmi** script.

Changing the NetWorker XBSA Environment

The **nsrdbmi** shell script contains several NetWorker XBSA environment variable settings shared by NetWorker and ON-Bar for backup tasks. The script is already configured to the default settings shown in Appendix A, “XBSA Environment Variables.” You can modify NetWorker XBSA settings shown in the script to any of the valid values shown in Appendix A.

To modify NetWorker XBSA variables not shown in the **nsrdbmi** script:

1. Copy the *nsrdbmi.sh* template to a new file and edit the new file with a text editor.
2. Add the NetWorker XBSA environment variable and the value you want assigned to it in the area of the script marked “Configuration variables, modifiable by user.”

To exclude logical logs from a scheduled or on-demand backup:

1. Copy the */etc/nsrdbmi.sh* template to a new file and open the new file with your favorite text editor.
2. Change the value assigned to DO_LOGFILE_BACKUPS and save the modified script.

   ```
   DO_LOGFILE_BACKUPS=NO
   ```

3. In the **nwadmin** Clients window for the NetWorker server, change the Backup command field for the OnLine Dynamic Server client to the name of the new script.

Changing the PATH Variable

The PATH environment variable must be configured to include the path to the ON-Bar executables and the NetWorker program **mminfo**. The PATH variable in the **nsrdbmi** script is already assigned the value:

```
$INFORMIXDIR/bin:/bin:/usr/sbin
```

If your ON-Bar executables are in a different path, change the value assigned to the PATH variable for $INFORMIXDIR in the **nsrdbmi** script.

Likewise, if your NetWorker executables are in a directory other than */usr/sbin*, be sure to change the value listed in the PATH variable assignment.

Changing the Value for INFORMIXDIR

The INFORMIXDIR environment variable defines where OnLine Dynamic Server is installed. The INFORMIXDIR variable in the **nsrdbmi** script is already assigned the value:

```
INFORMIXDIR=/usr/informix
```

If your OnLine Dynamic Server binaries are in a different path, change the value assigned to INFORMIXDIR in the **nsrdbmi** script.

To create a custom version of **nsrdbmi**:

1. Copy the **nsrdbmi** template, *nsrdbmi.sh*, to a new file.

   # **cp /etc/nsrdbmi.sh /usr/sbin/***mydbmi_script*

2. Use your favorite text editor and open the copied script to modify it.

Using Pre- and Post-Processing Commands

The **nsrdbmi** script contains two variables that allow you to use OnLine Dynamic Server commands to perform tasks before and after a scheduled backup.

PRECMD

The PRECMD variable allows you to customize your backup with a command to execute before the backup starts. For example, you could run a script that checks for corrupted dbojbects. If the script returns a nonzero value, the backup session fails and an error message indicates the failure.

POSTCMD

The POSTCMD variable allows you to customize your backup with a command to execute after the backup completes. For example, you could run a script that displays the status of your logical logs. If the script returns a nonzero value, an error message indicates the failure. Operator intervention may be required to return the database to a ready state.

Changing the Logfile Backup Setting

By default, NetWorker automatically backs up the logical logs associated with an OnLine Dynamic Server instance after completing a scheduled backup of the data. Once a logical log is successfully backed up, OnLine Dynamic Server automatically reuses the old logical log's space.

Caution: Logical log backups allow you to restore the database to its current state at the time a disaster occurs. If you do not maintain logical log backups and need to recover data, you will only be able to recover your database up to the time of the last backup. Transactions that occur between the last backup and the time your database is corrupted will be lost. Legato strongly recommends leaving DO_LOGFILE_BACKUPS enabled.

Chapter 3

Scheduled Backups

NetWorker backs up your OnLine Dynamic Server according to a schedule you set up. You add the OnLine Dynamic Server to the NetWorker server's list of clients and specify what you want backed up and how.

This chapter provides details on using NetWorker to set up the system running OnLine Dynamic Server as a client of the NetWorker server for scheduled and on-demand backups of OnLine Dynamic Server dbobjects. The script driving the connection between ON-Bar and NetWorker, **nsrdbmi**, is also described.

Be sure to read the *IRIX NetWorker Administrator's Guide*, shipped with your NetWorker server software, for complete information on using the NetWorker administrative program's graphical interface.

Customizing the nsrdbmi Script

When you installed DBMI on your OnLine Dynamic Server system, the installation placed a working copy of the **nsrdbmi** script in the directory where the NetWorker executables reside. A copy of the script, *nsrdbmi.sh*, was also installed in the */etc* directory for use as a template for developing custom DBMI scripts.

You may want to modify options in the **nsrdbmi** script, or you may want to have a custom script to control backups of different instances of OnLine Dynamic Server.

Caution: The **nsrdbmi** script must reside in the same directory on the system running OnLine Dynamic Server as the NetWorker **save** command. If you create a modified version of the script, keep the modified version in the same directory.

Chapter 4

On-Demand Backups

NetWorker DBMI provides scheduled, automated backup services for your OnLine Dynamic Server. The NetWorker XBSA library provided with DBMI also enables you to configure on-demand backups using the ON-Bar command-line interface. The command examples presented in this chapter are in C-shell format.

Performing On-Demand Backups Using ON-Bar

ON-Bar connects to NetWorker through an XBSA API, which passes dbobjects through a shared library. When you set up a system running OnLine Dynamic Server as a client of the NetWorker server, several variable settings are configured and passed to ON-Bar by the **nsrdbmi** script, which is invoked during a scheduled backup.

To perform an on-demand NetWorker backup using the ON-Bar command-line interface, you must first provide values for several required NetWorker XBSA environment variables. The environment variables can be set through the command line interface or a script in $INFORMIXDIR.

Caution: If you do not provide explicit settings for the required NetWorker XBSA variables, on-demand NetWorker backups default to the settings for a regular NetWorker filesystem backup, using NetWorker default configurations. This means that the log files, dbspaces, and blobspaces may be directed to the NetWorker "Default" media pool. See Appendix A, "XBSA Environment Variables" for the NetWorker XBSA environment variable default settings.

ON-Bar Backups and NetWorker Indexes

The NetWorker server's **savegrp** program invokes a level 9 backup of the client indexes and the NetWorker server's bootstrap save set each time a scheduled backup completes. The **savegrp** program is invoked only during a *scheduled* backup.

When you use the ON-Bar command line interface to perform an on-demand backup of OnLine Dynamic Server, the NetWorker client indexes for OnLine Dynamic Server and the NetWorker server's bootstrap save set are not backed up. If you *never* perform scheduled NetWorker backups, the NetWorker client indexes and NetWorker server's bootstrap save set are *never* backed up.

The client indexes and the NetWorker server's bootstrap save set are vital for restoring data to OnLine Dynamic Server in the event of a disaster. Performing regular, scheduled NetWorker backups of OnLine Dynamic Server provides maximum protection for your critical data.

Caution: If you *never* perform scheduled backups using NetWorker, you will not have backups of the client's indexes and server's bootstrap save set for use in the event of a catastrophic disaster.

Required NetWorker XBSA Variables for Data Backups

Substitute the values shown for the NetWorker XBSA variables in the following example with the same ones you assigned for scheduled NetWorker backups of OnLine Dynamic Server:

```
setenv NSR_SERVER networker_servername
setenv NSR_DATA_VOLUME_POOL dbmidata_pool
setenv NSR_LOG_VOLUME_POOL dbmilog_pool
```

Caution: If you do not specify a value for the NetWorker XBSA variable NSR_SERVER before issuing an ON-Bar command, NetWorker searches the network for the correct server to use. Setting the NSR_SERVER variable helps avoid a potential delay in the backup process.

If you do not have a NetWorker configuration for scheduled backups of OnLine Dynamic Server, you can substitute the values for the required NetWorker XBSA variables with any valid NetWorker XBSA value. See Appendix A, "XBSA Environment Variables" for information about the NetWorker XBSA environment variables and their valid values.

Note: You must be logged in as *informix* or *root* to use ON-Bar commands for on-demand NetWorker backups of OnLine Dynamic Server dbspaces, blobspaces or logical log files.

Example of an On-Demand Backup Command

The commands shown in the following example

- perform a level 0 (NetWorker “full”) backup of the dbobject “dbspace01” on an OnLine Dynamic Server instance named “venus”
- back up all full logical log files associated with “dbspace01”
- close the current logical log
- back up the closed logical log

```
setenv NSR_SERVER jupiter
setenv NSR_DATA_VOLUME_POOL DBMIData
setenv NSR_LOG_VOLUME_POOL DBMILogs
onbar -b -L 0 dbspace01
onbar -l -c
```

NetWorker XBSA translates the instance’s dbobject “dbspace01” and the level 0 specification into “INFORMIX:/venus/dbspace01/0” for the save set entry in the NetWorker server’s online file and media indexes. The last field in the save set entry for a manually backed-up dbobject indicates the backup level.

For details on using ON-Bar commands, and backup strategy suggestions for protecting OnLine Dynamic Server data, refer to “Using ON-Bar” in the *INFORMIX–Online Dynamic Server Backup and Restore Guide* provided with OnLine Dynamic Server.

Performing Continuous Logical Log Backups

ON-Bar is configured, by default, to back up logical logs automatically once the log file is filled. After the log file is successfully backed up, ON-Bar closes the file, frees the space used by the file, and opens a new file for transaction logging. Log file backups are always performed as a “full” backup (ON-Bar level 0).

Caution: For continuous log backups, Informix recommends dedicating a backup device to the logical log backup process. This ensures that a device on the backup server is always available to receive logical log data.

Required NetWorker XBSA Variables for Continuous Log Backups

If you choose to maintain continuous backups of your logical logs, modify the OnLine Dynamic Server *$INFORMIXDIR/etc/log_full.sh* script to include the NetWorker XBSA variables. Assign the same values used for scheduled NetWorker backups of the log files associated with the OnLine Dynamic Server instance:

```
NSR_SERVER = networker_servername
NSR_LOG_VOLUME_POOL = dbmilog_pool
export NSR_SERVER
export NSR_LOG_VOLUME_POOL
```

If you created a customized **nsrdbmi** script and modified values for other NetWorker XBSA variables, be sure to include the modified variables and assigned values in the *$INFORMIXDIR/etc/log_full.sh* script. Otherwise, the NetWorker XBSA variables default to the values described in Appendix A, "XBSA Environment Variables."

Note: Even if you do not use logging for OnLine Dynamic Server, a log file containing administrative information such as checkpoint records and additions and deletions of chunks is maintained. Backups of this type of log file are very small. Even if you do not use logging, maintaining backups of the abbreviated log file are required to restore data from dbspace backups. The only type of backup that can be restored without logs is a whole-system backup.

Chapter 5 provides information on restoring backed up data to OnLine Dynamic Server.

Chapter 5

Restoring Data

DBMI provides a means for ON-Bar to restore data backed up by NetWorker to your OnLine Dynamic Server in the event of data corruption or a disk crash. DBMI allows you to recover OnLine Dynamic Server instances and individual objects on-demand, using the ON-Bar command-line interface.

Use the information in this chapter in conjunction with the sections "Syntax for Restoring Data" in the *INFORMIX–Online Dynamic Server Backup and Restore Guide*, "Recovering from a Disk Crash" in the *IRIX NetWorker Administrator's Guide*, and the nsr_crash(1M) reference page. The command examples presented in this chapter are in C-shell format.

Preparing to Restore Data

During a scheduled or on-demand backup, the NetWorker server makes an entry in an online client index and records the location of the data in an online media index. These entries provide recovery information needed for every OnLine Dynamic Server object backed up. The client index entry is maintained in the index until the browse policy configured for the client's save set expires. The media index entry is maintained until the retention policy configured for the client's save set expires.

Once the browse and retention policies expire, the backup media is eligible for recycling and may be overwritten. Until the media is relabeled, the data is still recoverable, using the NetWorker **scanner** command. The scanner(1M) reference page provides complete details on recovering save sets whose client index entries have expired.

To view online client index entries, click the Indexes speedbar button in the NetWorker administration program's main window and select an entry to view. The listing displayed for the entry you select shows the save set ID assigned during a backup session, the number of files and their size, and the date and level of the backup session.

NetWorker sends a record of the server's *bootstrap* save set generated during the backup to your default printer, so you have a printed record of the dates, locations, and save set ID numbers for the server's online indexes needed for recovering data. Keep the

bootstrap printouts in a file for quick reference in the event a disaster, such as a disk crash or server malfunction, occurs.

Restoring Data with ON-Bar

You use ON-Bar commands to restore data backed up from a scheduled or on-demand NetWorker backup. The shared XBSA library translates OnLine Dynamic Server instance names passed by ON-Bar into NetWorker save set names for retrieval from the NetWorker server's online client index and restoration to OnLine Dynamic Server.

Caution: If you do not specify a value for the NetWorker XBSA variable NSR_SERVER before issuing an ON-Bar command, NetWorker searches the network for the correct server to restore data from. Setting the NSR_SERVER variable helps avoid a potential delay in the restore process.

What Types of Restores Can ON-Bar Perform?

ON-Bar supports several types of database and dbobject restores from the backup media managed by NetWorker:

- Physical

  ```
  setenv NSR_SERVER networker_servername
  onbar -r -p [dbspace_name]
  ```

  Physical restores replace lost or corrupted OnLine Dynamic Server dbobjects from NetWorker backup media. A physical restore can be performed as a whole-system or selected-dbspace restore.

- Logical

  ```
  setenv NSR_SERVER networker_servername
  onbar -r -l
  ```

  Logical restores recover the server transactions made since last dbobject backup, followed by a rolling forward of the logical logs backed up for the dbobjects. If different backup sessions are involved, the log rolls forward transactions made since the backup time recorded for each dbobject restored.

- Combined

  ```
  setenv NSR_SERVER networker_servername
  onbar -r [dbspace_name]
  ```

Combined restores allow you to issue a single command to perform a physical restore immediately followed by a logical restore.

- Point-in-time

  ```
  setenv NSR_SERVER networker_servername
  onbar -r -t time -w -p
  ```

  Point-in-time restores involve performing a whole-system, physical restore of OnLine Dynamic Server data from a whole-system backup to a specified time instead of the default, which is the last time of OnLine Dynamic Server backup.

- Imported

  ```
  setenv NSR_SERVER networker_servername
  setenv NSR_CLIENT networker_clientname
  onbar -r -w
  ```

  Imported restores allow movement of data from one OnLine Dynamic Server instance to another. The versions of XBSA must be compatible between the OnLine Dynamic Server instances involved.

  Caution: To perform an imported restore, you must first set the Remote access field in the NetWorker Clients window for the OnLine Dynamic Server instance to allow a different OnLine Dynamic Server instance to access the backed-up data, if the instances reside on different hosts.

  Before you perform an imported restore, do the following:

  - Perform a whole-system backup on the OnLine Dynamic Server you are importing data from—you cannot transfer specific dbobjects.
  - Set up the new OnLine Dynamic Server system with exactly the same disk layout as the one you are transferring from.
  - Copy the ONCONFIG file from the existing system to the new system.
  - Change the *servername* and *servernumber* entries specified in the new system's ONCONFIG file.
  - Copy the emergency boot file from the existing system to the new system, renaming the file *ixbar.new_servernumber.*
  - Copy the *sqlhosts* file from the existing system to the new system.
  - Copy the *oncfg_servername.servernumber* file from the existing system to the new system.

  For complete details on performing an imported restore, refer to the *INFORMIX–Online Dynamic Server Backup and Restore Guide.*

OnLine Dynamic Server Restore Modes

You perform three types of restores with OnLine Dynamic Server:

- Cold, which consists of a physical and logical restore of the critical dbspaces, then a physical and logical restore of the noncritical dbspaces. A cold restore is performed with the OnLine Dynamic Server in off-line mode. After a cold restore completes, OnLine Dynamic Server is left in "quiescent" mode.

  Note: A cold restore of selected dbspaces succeeds only if the critical dbspaces are included on the restore command line. Critical dbspaces are defined as the root dbspace and any dbspace containing either physical or logical logs. Refer to the *INFORMIX*–OnLine Dynamic Server *Backup and Restore Guide* for further details on performing cold restores of OnLine Dynamic Server.

- Warm, which consists of one or more physical restores, a closing and backup of the current logical log, followed by a logical log restore. A warm restore is performed with the OnLine Dynamic Server in on-line or quiescent mode.

- Mixed, which consists of a cold restore of the critical dbspaces, with OnLine Dynamic Server in off-line mode followed by a warm restore of noncritical dbspaces, with OnLine Dynamic Server in quiescent or on-line mode. Using a mixed restore allows you to quickly recover critical dbspaces, plus any data that users require immediate access to. Once OnLine Dynamic Server is returned to quiescent mode, you can perform a warm restore of the other dbobjects.

Disaster Recovery

Hardware malfunctions rarely occur at convenient times, but if you have a disaster recovery plan in place and use DBMI to maintain regular backups of your OnLine Dynamic Server instances and the logical logs associated with them, you are well-equipped to recover critical data in a timely manner.

The information presented in this section presumes that you have read and are familiar with the procedures outlined in your OnLine Dynamic Server documentation and the information presented in the section "Recovering from a Disk Crash" in the *IRIX NetWorker Administrator's Guide.*

OnLine Dynamic Server Disk Crash

If the primary disk containing critical OnLine Dynamic Server dbobjects and NetWorker client-side binaries is damaged, do the following:

1. Reinstall the NetWorker client binaries, the DBMI software, and the OnLine Dynamic Server software, if needed. If you perform regular NetWorker backups of your system binaries, you can use NetWorker to recover the system data.
2. Use NetWorker to recover the emergency boot file and configuration file for the OnLine Dynamic Server instance.

   ```
   recover -t date -a \
   /usr/informix/etc/sqlhosts \
   /usr/informix/etc/onconfig.std \
   /usr/informix/etc/ixbar.servernum \
   /usr/informix/etc/oncfg_servername.servernum
   ```

3. If the physical media containing the logical logs must be replaced before beginning the restore, manually salvage the current logical log file.

   ```
   onbar -l -s
   ```

4. Use ON-Bar to restore data from the most recent backup.

   ```
   onbar -r
   ```

   Once the restore completes, OnLine Dynamic Server is left in quiescent mode.

NetWorker and OnLine Dynamic Server Disk Crash

If the NetWorker server's primary disk containing the online indexes (the */nsr* filesystem) and the primary disk for OnLine Dynamic Server are both damaged:

1. Reinstall the NetWorker server binaries, if needed.
2. Find the latest bootstrap printout for the NetWorker server and follow the procedure outlined in "Recovering from a Disk Crash" in the *IRIX NetWorker Administrator's Guide* to recover the server's online indexes.

   Caution: Do not attempt to recover the NetWorker server's online file or media indexes to a different directory than the one they were backed up from. Once you recover the indexes to their original location, you can safely move them to another directory. Refer to the *IRIX NetWorker Administrator's Guide* for details on moving the indexes.

3. Reinstall the DBMI software and OnLine Dynamic Server, if needed.
4. Use NetWorker to recover the emergency boot file and configuration file for the OnLine Dynamic Server instance.

   ```
   recover -t date -a \
   /usr/informix/etc/sqlhosts \
   /usr/informix/etc/onconfig.std \
   /usr/informix/etc/ixbar.servernum \
   /usr/informix/etc/oncfg_servername.servernum
   ```

5. If the physical media containing the logical logs must be replaced before beginning the restore, manually salvage the current logical log file.

   ```
   onbar -l -s
   ```

6. Use ON-Bar to restore data from the most recent backup.

   ```
   onbar -r
   ```

   Once the restore completes, OnLine Dynamic Server is left in quiescent mode.

Refer to the *INFORMIX–Online Dynamic Server Backup and Restore Guide*, provided with your OnLine Dynamic Server software, for further information on using ON-Bar to restore data from backup media managed by NetWorker.

Appendix A

XBSA Environment Variables

This appendix gives the steps for changing a NetWorker XBSA environment variable, and lists those variables, their default values, and valid options.

NetWorker XBSA

The NetWorker XBSA allows for configuration of environment options to activate certain features of NetWorker not directly supported by X/Open specifications. NetWorker XBSA enables ON-Bar and NetWorker to interact during backups and restores.

- The interaction ensures restoration of dbobjects to the correct OnLine Dynamic Server instance and to their proper sequence in the database.
- NetWorker XBSA compiles a history of objects backed up for the OnLine Dynamic Server instance.

See Appendix B, “Error Messages” for a description of error messages associated with a NetWorker XBSA session.

Caution: NetWorker and ON-Bar rely on shared XBSA libraries for interaction during backups and restores. Create a symbolic link, using the appropriate shared library file extension for your system. In the following example, which creates a symbolic link between the NetWorker and ON-Bar XBSA libraries, “so” indicates the file extension for IRIX systems.

```
ln -s /usr/lib/libbsa.so /usr/lib/ibsad001.so
```

Changing NetWorker XBSA Variables

To change a value for a NetWorker XBSA variable that does not appear in the default **nsrdbmi** script:

1. Copy the template file */etc/nsrdbmi.sh* to the directory where the NetWorker binaries are installed.
2. Add the NetWorker XBSA environment variable to the script and assign a valid value to the variable.
3. Add the NetWorker XBSA environment variable to the list of variables exported.

```
export_environment_variables()
{
    export NSR_VOLUME_POOL
    ...
    export new_NSR_variable
}
```

4. Save the edited script with a descriptive filename.
5. Edit the NetWorker server's client resource. For example, in the **nwadmin** Clients window for the OnLine Dynamic Server instance, enter the filename of the custom **nsrdbmi** script in the Backup command field.

   Caution: If you choose continuous log file backups or perform an on-demand backup of a dbobject, the default settings for the NetWorker XBSA variables will override values set with the NetWorker administration program. See Chapter 4 for recommendations about NetWorker XBSA variables that you need to assign explicit values for.

Default Values and Valid Options

This section contains tables for each of the NetWorker XBSA environment variables you can modify by customizing the **nsrdbmi** script. Most of the environment variables described appear in the default **nsrdbmi** script, while others are set in the NetWorker XBSA shared libraries included with Database Module for Informix.

NSR_BACKUP_LEVEL

Definition	The NSR_BACKUP_LEVEL environment variable indicates the NetWorker backup level to use for the XBSA session.
Default Value	full
Possible Values	All valid NetWorker levels are supported. The 12 possible values for NetWorker levels are: **full**, **1** through **9** (indicated by the integers 1 through 9), **incr**, and **skip**. If you specify the **skip** level for a save set, the savegroup completes successfully, but does not perform a backup of the data. **Note**: While NetWorker supports incremental backups of levels 1 through 9, ON-Bar only supports incremental levels 1 and 2. If you assign a level other than skip, full, 1, or 2, the backup fails and displays the message “Only level 0 (full), 1, or 2 backups allowed.”

NSR_CLIENT

Definition	The NSR_CLIENT environment variable indicates the NetWorker client resource to use for the XBSA session.
Default Value	Host from which the XBSA session is initiated, as indicated by **getlocalhost()**.
Possible Values	Since the client name is an arbitrary string, the value for NSR_CLIENT is not checked directly. An incorrect value may cause an authentication or system error in NetWorker.

NSR_COMPRESSION

Definition	The NSR_COMPRESSION environment variable indicates whether to compress the backup data as it is sent to the NetWorker server.
Default Value	FALSE
Possible Values	Setting NSR_COMPRESSION to a value of TRUE means the standard compression technique for XBSA for NetWorker is performed on the data backed up. Setting NSR_COMPRESSION to a value of FALSE means that compression is not performed. **Note**: Compressing data from the database server may speed backups, as long as the database server is able to send data to the backup server fast enough to keep the tape drive streaming. Data compression during backup will impact CPU usage on the backup server, but will reduce the amount of data sent to the NetWorker server.

NSR_DATA_VOLUME_POOL

Definition	The NSR_DATA_VOLUME_POOL environment variable indicates the volume pool to which data files should be backed up.
Default Value	XBSA does not set a pool by default—if none is specified, the pool is selected by the NetWorker server based on configuration of the Pools resource.
Possible Values	Any valid NetWorker pool name of 1024 characters or less. This value is set in the NetWorker pools resource and can be explicitly assigned in a shell script.

NSR_DEBUG_FILE

Definition	The NSR_DEBUG_FILE environment variable indicates the full pathname and filename where NetWorker XBSA messages should be written. Message logs for XBSA are separated from regular NetWorker messages. NetWorker XBSA error messages are indicated by the prefix “BSA.”
Default Value	**/nsr/applogs/xbsa.messages**
Possible Values	Any valid values for pathname and filename are valid. If the file specified cannot be opened, a BSA_RC_INVALID_KEYWORD error message is written to one of the following locations: • the alternate messages directory created during installation, */nsr/applogs* • the directory assigned to the TMPDIR environment variable • the */tmp* directory, if TMPDIR is not set

NSR_DEBUG_LEVEL

Definition	The NSR_DEBUG_LEVEL environment variable allows you to set the level of NetWorker XBSA error report messages sent to the *xbsa.messages* log file.
Default Value	The default value is **2**, which means that critical error messages and all network (RPC) errors are written to *xbsa.messages*.
Possible Values	Any integer from 0 to 9 is valid. Higher values generate more detailed reports: • A value of **0** means that no error messages are written to *xbsa.messages*. • A value of **1** means that only critical error messages are written to *xbsa.messages*. • A value of **2** means that all network (RPC) errors are written to *xbsa.messages*. • A value of **3** means that all NetWorker XBSA informational messages are written to *xbsa.messages*.

NSR_GROUP

Definition	The NSR_GROUP environment variable indicates which backup group to use for a backup session.
Default Value	None
Possible Values	Any valid NetWorker group name of 1024 characters or less. Invalid group names may cause authentication or system errors in another routine. A NetWorker save group acts as an “alarm clock,” notifying the NetWorker server that a group of clients have a backup scheduled to occur at the time designated in the group’s “start time” attribute.

NSR_LOG_VOLUME_POOL

Definition	The NSR_LOG_VOLUME_POOL environment variable indicates the volume pool to which logical logs should be backed up.
Default Value	NetWorker XBSA does not set a pool by default—if none is specified, the pool is selected by NetWorker based on configuration of the Pools resource.
Possible Values	Any valid NetWorker pool name of 1024 characters or less. This value is set in the NetWorker pools resource and can be explicitly assigned in a shell script.

NSR_NO_BUSY_ERRORS

Definition	The NSR_NO_BUSY_ERRORS environment variable indicates whether the savegroup should wait for a busy NetWorker server or fail immediately.
Default Value	FALSE. Wait for the NetWorker server to accept the connection.
Possible Values	Setting NSR_NO_BUSY_ERRORS to a value of TRUE causes the backup to fail immediately when the NetWorker server is busy. A network error message describing the reason for the failure is written to *xbsa.messages*. If NSR_DEBUG_LEVEL is set to **1** and the NetWorker server is busy, the backup process stops and the error message “BSA_RC_ABORT_SYSTEM_ERROR” appears in *xbsa.messages*.

NSR_PROCESS_ENVIRON

Definition	The NSR_PROCESS_ENVIRON variable indicates whether the environment vector for the process calling NetWorker XBSA should be processed along with the explicit XBSA environment vector.
Default Value	TRUE. Process environment vectors for the process calling XBSA. Environment options passed from XBSA take precedence over options set by the calling process.
Possible Values	FALSE. NetWorker XBSA will ignore the environment vectors passed from the calling process (for example, the shell that the **onbar** command was executed from).

NSR_SAVESET_NAME

Definition	The NSR_SAVESET_NAME environment variable indicates the save set name NetWorker XBSA should use for a save session.
Default Value	If NSR_SAVESET_NAME is not assigned a specific value, NetWorker XBSA will use the format "INFORMIX:/*objectSpaceName*," where *objectSpaceName* is the name of the OnLine Dynamic Server instance.

Possible Values	Any valid NetWorker save set name. Only the first 63 characters are meaningful to the NetWorker server's media database.
	For dbspaces and blobspaces:
	`INFORMIX:/objectSpaceName/spacename/level`
	For logical logs:
	`INFORMIX:/objectSpaceName/logspace/log_number`

NSR_SERVER

Definition	The NSR_SERVER environment variable indicates the hostname of the server NetWorker XBSA should use for a save session.
Default Value	The most appropriate server, based on the index name and client name for the session. *See also* NSR_CLIENT
Possible Values	The server name defined by the NSR_SERVER environment variable is checked, using **gethostbyname**(). If this routine call fails, the NetWorker XBSA error code "BSA_RC_INVALID_KEYWORD" is returned.

Appendix B

Error Messages

This appendix lists error messages you may encounter while using Database Module for Informix and provides suggestions to resolve the problems described.

ON-Bar Messages

When ON-Bar encounters an error or condition requiring a warning, it writes a message to the assigned message file. The default message file for ON-Bar is */tmp/bar_act.log*. Refer to Appendix A in the *INFORMIX-OnLine Dynamic Server Backup and Restore Guide* for a listing of ON-Bar messages.

NetWorker Messages

NetWorker error messages are displayed in the **nwadmin** window. The Messages display lists all messages generated during the past 24 hours. Error messages are also written to */nsr/messages/daemon.log.*

NetWorker error messages appear in the format

`day hh:mm:ss daemon_or_program_name: message`

Errors Messages Generated While Saving Data

This section lists error messages encountered during a NetWorker backup. The messages are organized alphabetically by NetWorker daemon name and program name to make them easier to match to the NetWorker message displayed.

nsrck

`cannot lock flag file for` *clientname*`:` *reason*

The flag file signifying the end of the first part of index compression is already in use by another instance of the **nsrck** program, or by the **nsrindexd** daemon. Since disaster will ensue if two processes access the same index at the same time, **nsrck** will refuse to act on the named file.

`checking index for` *clientname*

The files associated with the named client are being inspected.

`completed checking` *count* `clients`

As the program finishes, this indicates some form of checking was accomplished.

`compressing index for` *clientname*

The **-x** or **-C** (compress) option has taken effect.

`cross-checking index for` *clientname*

The **-X** (cross-check) option is in effect.

`more space needed to compress` *clientname* `index,` *size required*

The **nsrck** program cannot find enough disk space to hold the temporary file *db.CMP*. The operator should free some disk space on any local filesystem and retry the command. The **df** command may be used to see how much free space is available on any filesystem.

`rolling forward index compression for` *clientname*

After a reboot, if index compression completed its first copy, the compression is rolled forward.

`WARNING no valid savetimes - cross-check not performed for` *clientname*

During a cross-check, no save sets were found for this client. Since this situation can occur during disaster recovery, **nsrck** avoids deleting the entire contents of the client index and instead does nothing.

nsrexecd

`/path/nsrexecd: Can't make pipe`
`/path/nsrexecd: Can't fork`
`fork: No more processes`

The specified client-side resource has been exceeded. There are too many other services running on the client while **savegrp** is running. Inspect the client and determine why it has run out of resources. The client may need to be rebooted. You should also consider rescheduling any jobs automatically started on the client (for example. via **cron**) that run while **savegrp** is running.

`/path/nsrexecd: Couldn't look up address for your host`
`/path/nsrexecd: Host address mismatch for server`

The **nsrexecd** daemon on the client managed to look up the server in the client's host table, but the address listed there did not match the address of the server. Every interface of the server must have a unique name listed in the host table (possibly with non-unique aliases or CNAMEs), and each unique name must be listed as a valid server to **nsrexecd**.

`/path/nsrexecd: Host server cannot request command execution`
`/path/nsrexecd: Your host cannot request command execution`

The server is not listed in nsrexecd's list of valid servers on the specified client. The list of valid servers is either on the **nsrexecd** command line (with one or more **-s *server*** options to **nsrexecd**), or in a file (with the **-f *file*** option to **nsrexecd**). It may also be the case that the server is not listed in one or more of */etc/hosts*, NIS, or DNS, depending on the client, in which case **nsrexecd** cannot validate the server until the client's host naming configuration is fixed.

`/path/nsrexecd: Invalid authenticator`
`/path/nsrexecd: Invalid command`

These two messages should never occur in a savegroup completion message. They mean that **savegrp** did not follow its protocol correctly.

`/path/nsrexecd: Permission denied`
`Permission denied`

These similar messages are generated by **nsrexecd** and **rshd**, respectively. In either case, the server does not have permission to execute commands on the client. In the case of the first message, make sure that the server is listed as a valid server on the client (see "/*path*/nsrexecd: Host server cannot request command execution" for details). In the case of the second message, which does not mention **nsrexecd**, make sure that "servername" is listed in the client's */.rhosts* file (or, if you have set the remote user attribute for this client, the *.rhosts* file in the home directory for that user on the client).

`Login incorrect`

The remote user attribute for the client is not set to a valid login on the client. Verify that the remote user attribute for the client is set to the correct login name. You may see this message even when running **nsrexecd** if **nsrexecd** has not been started (or was killed) on the client.

`socket: protocol failure in circuit setup`

The client does not seem to support the TCP/IP protocol stack, or has not used a privileged port for setting up its connection. The latter could occur if you use **nsrexecd** but did not start it as *root* on the specified client. The **nsrexecd** daemon must run as *root* on each client.

nsrindexd

`lock on filename acquired`

The advisory lock that the daemon was waiting to clear has been cleared; see "waiting for lock" message.

`waiting for lock on filename`

Another program is accessing the same file that is required by the **nsrindexd** daemon. The daemon will wait for the advisory lock to be cleared.

nsrmmdbd

`A copy of this process is already running!`

Another copy of **nsrmmdbd** is currently running and has exclusive access to the media database. Only one **nsrmmdbd** process should be running on a given machine at a time. This can happen if the previous **nsrmmdbd** was not properly killed off. Use **nsr_shutdown** or **ps** and **kill** to identify and kill off all the NetWorker daemons before restarting **nsrd** again.

`Cannot open lock file`

This is an internal error; check permissions on the */nsr/tmp* and */nsr/mm* directories.

`media db is saving its data, this may take a while`

The daemon is dumping its records to a temporary file while the database is being backed up. The service is unavailable while the database is dumping.

`media db is recovering, this may take a while`

The **nsrmmdbd** daemon is reloading its database. The service is unavailable while the data is being reloaded.

`media db is cross checking the save sets`

Printed each time the daemon is restarted. On start-up, the daemon sanity checks its records before providing its service.

`media db is open for business`

After any of the previous messages, this message is printed to indicate that the service is once again available.

save — savefs

`Access violation from` *client* `- insecure port` *N*

This message, generated by the **save** command on client, means that **save** is not **setuid** *root*. Make sure that the **save** command on the client is owned by *root* and has its **setuid** bit set. If **save** is on an NFS mounted filesystem, make sure the filesystem was not mounted on that client using the `-nosuid` option.

asm: chdir failed */path*: Permission denied

While backing up the specified save set, **save** was unable to enter the named directory. This may mean that **save** is not **setuid** *root* on the specified client, or that the directory is actually an NFS mount point for which *root* is not allowed access. Check the permissions for **save** on the specified client (using **ls**) and make sure that **save** is owned by *root* and that the **setuid** bit is set.

RPC error, details...Cannot open save session with *'server'*

The **save** command generates this message if it is unable to back up data to the NetWorker server. There are several possible detailed reasons. The most likely causes are: resources are exceeded on the server so **nsrd** cannot accept new **save** sessions, **nsrd** actually died since **savegrp** started (however, this is unlikely, since you cannot normally receive a **savegrp** completion message after **nsrd** dies, but you can see this when using the **-p** option), there are numerous network errors occurring and **save** cannot open a session to save its data (check this by running **netstat -s** and see how many network errors are occurring; you may need to do this several times a few minutes apart to get the change in errors). Save cannot tell which of these three causes are the real cause. If you see these errors frequently, and it looks like a server resource problem, you might consider increasing the value of the "client retries" attribute of the group resource having these problems. This won't decrease the resource utilization, but will make **savegrp** more robust (the trade-off is that increasing client retries will increase the load on the server even more).

save: *client.xxx.com* is not on *client*'s access list

This error occurs when the named client has more than one name, for example, a short name, *client*, and a fully-qualified domain name, *client.xxx.com*. When the client attempts to connect back to the NetWorker server to start a **save**, that client is calling itself by the name *client*, which matches the client resource name, but when the server looks up the client's network address, it is getting back the name *client.xxx.com*. If this is, in fact, correct, add the name *client.xxx.com* to the client's aliases attribute, and rerun the **save**.

`save: path length of n too long, directory not saved`

This message can occur if you have a directory tree that is very deep, or directory names that are very long. This message can also occur if there are bad blocks in the specified filesystem, or if the filesystem is corrupt. NetWorker limits the full pathname to 1024 characters, which is the system imposed maximum on most systems. To save such directories, you need to rename or move the directories so that the full pathname is shorter than 1024 characters. If the filesystem appears to be corrupted (for example, a very long pathname that looks like it has a loop in the name), perform a filesystem check on the specified client.

/path`/savefs: Command not found`
/path`/save: Not found`
/path`/save: Command not found`
/path`/savefs: Not found`

The **save** or **savefs** command could not be found in the specified path. If you are using **nsrexecd**, this probably means that the **save** or **savefs** command is not in the same directory in which **nsrexecd** is installed (or that **save** or **savefs** was removed). If you are using **rshd** for remote execution, then you need to set the executable path attribute in the Client resource for this client to be the directory in which the NetWorker executables are installed on the client.

`savefs: error starting save of` *filesystem*

This message accompanies several other **save** or *asm* messages listed, and means that **savefs** has detected the failed **save** and has marked the save set as failed.

save`: unknown host name: server`
savefs`: unknown host name: server`

The host table on the specified client (either */etc/hosts*, NIS or DNS, depending on that client's configuration) does not include the server's name. You need to add the server's hostname to the specified client's host table. Note that if you use DNS but the server's Client resource name (in other words. the client resource for the server itself) is not fully qualified (i.e. it looks like "server," not "server.domain," and the server is in a different domain from the client, you will need to add the name server to the domain table for the domain containing the client. If you use NIS, this error means that either the NIS hosts map does not contain the server, the */etc/hosts* file does not list the server, or the NIS master for the specified client is otherwise misconfigured (the server is a secondary server and there is no **yppush** from the primary; run **ypwhich -m** on the client to find out which NIS server is providing master translation).

`unknown host`

The specified client is not listed in the host table on the server (similar to "Warning: 'client' is not in the hosts table!"). Depending on your host configuration, this means the client is not listed in one (or more) of */etc/hosts*, NIS, or the DNS. If you use fully qualified domain names, you may need to make a new client resource for this client, using that fully qualified domain name (i.e. name the client resource *mars.acme.com*, not *mars*).

`Warning: 'client' is not in the hosts table!`

This message is generated by a **save** or **savefs** command run on the specified client to save that client's filesystems. The client's hostname is not listed in the host table on the client (either */etc/hosts*, NIS or DNS, depending on that client's configuration). This almost always results in a failed **save**. Fix the client's host table and rerun **save**.

`Warning – file 'path' changed during save`

This warning message is generated when **save** notices that the file's modification time changed while the file was being backed up. NetWorker does not attempt to lock files before saving them, since this would make backups run extremely slowly. You may wish to backup files that generate this message manually, to ensure that a consistent copy is saved. NetWorker does not reattempt backup automatically, so as to avoid trying forever on the same file.

`save: path file size changed!`

This informational message is often generated when NetWorker backs up the message log files. It may also occur for other files. For files that you expect to grow while **savegrp** is running, you can use a directive specifying that the **logasm** (see the uasm(1M) reference page) should be used to back up the file. *See also* the nsr(4) and nsr_directive(4) reference pages.

`save: network error, server may be down`

The backup of the named filesystem was begun, but the connection to the NetWorker server closed part way through. This typically means that the server machine rebooted, or one or more **save** agents were killed by the system administrator or by the system itself (for example. due to overwriting the binary or a disk error in swap space). Restart **save** at a later time.

savegrp

`Aborted`

This informational message occurs only when you stop a running **savegrp**. The session for this save set may not disappear immediately, especially if the program's attempt to kill the save session fails. When you restart the **savegrp** command, it retries the discontinued save set.

`Access violation – unknown host:` *client*

The client's hostname and IP address are not correctly listed in one or more of */etc/hosts*, NIS, or DNS on the server. You need to either change the appropriate host table (depending on which one(s) are in use on your server) to list the client's name as it is known to NetWorker, as that client's primary name, or you need to add the name listed at the end of the error message to the aliases attribute of the client's Client resource(s).

asm`: cannot open /`*path*`: I/O error`

This message generally means that there are bad blocks on the disk(s) containing the specified file or directory. You should immediately run a filesystem check on the named client filesystem and check your client's system error log. If there are bad blocks, repair them if possible, or move the filesystem to a different disk.

asm`: cannot stat /`*path*`: Stale NFS file handle`
asm`: cannot stat /`*path*`: Missing file or filesystem`

These informational messages (or variants of them for other operating systems) mean that the when save attempted to test the named directory to determine if it was a different filesystem from the one currently being saved, the filesystem was, in fact NFS mounted, but the mount point was bad. While this message does not affect the saved data, it does mean you have a network or NFS problem between the specified client and one or more of its fileservers. You may need to remount filesystems on the client, or perhaps reboot it, to correct the problem.

asm`: external ASM '`*asm2*`' exited with code 1`

This message generally accompanies another message reporting a specific problem encountered while saving a file or directory on the named save set. The backup will attempt to continue and attempt to save other data, and generally, the backup will not be listed in the failed save sets section of the completion mail if any files on the save set are saved successfully, even if it only saves the top directory of the save set.

asm: getwd failed

While **savegrp** was backing up the specified save set, an attempt to determine the current directory's name failed. This occurs on clients, generally running older versions of the NetWorker ClientPak, on which the getwd(3) library call is broken. You may want to contact Legato Technical Support to find out if there is a patch available for your client platform to work around this vendor-specific bug, or contact your operating system vendor to see if a more recent OS version addresses this problem.

asm: missing hard links not found

A backed-up file had one or more hard links that were not found. The message is followed by a list of one or more filenames that were backed up minus some links. The message means that the files were either created (with multiple hard links) while the backup was occurring, so some of the links were missed due to the order of filesystem tree walking, or the file (or some links) was removed while the backup was occurring. Only those links that were found can be recovered; additional links will have been lost. You can do an additional incremental backup of the affected filesystem if a consistent state for the affected file is essential.

asm: */path* was not successfully saved

This message generally accompanies one or more other more-specific messages for the save set. The specified path within the current save set was not saved successfully. The backup will continue trying to back up other files and directories on the save set.

asm: xdr_*op* failed for */path*

This error can be caused by several possible conditions (out of memory, buggy networking software in the operating system, an external ASM unexpectedly exiting, or a lost network connection). If it was due to a lost network connection, the NetWorker server most likely exited (due to **nsr_shutdown**). After restarting the server, rerun the group. If due to an ASM exiting unexpectedly (in this case, the message should be accompanied by a message describing which ASM exited unexpectedly), you may have found a bad block on the disk, or perhaps a bug. Check if the client ran out of memory (there may be console messages), and verify that there are no bad blocks on the save set's disk. If there were network errors, there may also have been messages logged by other programs on the system console (client or server), or to system log files.

`connect to address` *AA.BB.CC.DD*`:` *message*
`Trying` *AA.BB.CC.DD*`...`

These informational messages are displayed only when the **-v** option is used. They mean that the connection to the client failed on the address specified in the first line of the message. If the client has more than one IP address, **savegrp** has attempted the address listed in the second line. Looking at subsequent lines of the savegroup completion notice may show if this second address succeeded. You may want to check and change your network routing tables to avoid getting these messages.

`Connection refused`

The client machine is up, but it is not accepting new network connections for **nsrexecd** (or **rshd**). This could mean the client was in the process of booting when **savegrp** attempted to connect, or that the client had exceeded some resource limit, and was not accepting any new connections. You should attempt to log into the client and verify that it is accepting remote connections.

`Connection timed out`

This usually means the client has crashed or is hung. Make sure the client has rebooted, and that **nsrexecd** is running on it (if you are using **nsrexecd**).

`group` *groupname aborted, savegrp is already running*

This message is delivered by itself. It occurs when the named group has already been started or restarted (after a reboot, or when requested via the Group Control Window of **nwadmin**), either automatically by **nsrd** or manually from the command line. You can use **ps** to find out the process ID of a running **savegrp**. The existence of a running group is determined by looking for a file named */nsr/tmp/sg.groupname* which, if existing and locked, means **savegrp** is running.

`has been inactive for` *n* `minutes since` *time*.
client:*saveset* `is being abandoned by savegrp.`

A backup of the specified save set started, but after *N* minutes of no activity, **savegrp** gave up on the save set. Generally, this means that the client is hung waiting for an NFS partition. Unfortunately, NetWorker (or any other program) has no reliable way of telling if an NFS partition will hang until after it tries to access the partition. When the partition comes back on line, the **save** will complete, despite the fact that **savegrp** abandoned it. You should check the client, however, since you sometimes need to reboot the client to unhang NFS partitions. Non-UNIX clients also hang for other reasons, most notably, bugs in the operating system implementation of their network protocols.

`Host is unreachable`

The NetWorker server cannot make TCP/IP connections to the client. This generally means the network itself is not configured correctly; most commonly, one or more gateways or routers are down, or the network routes were not set up correctly. You should verify that the server can connect to the client. If the server cannot connect to the client, check your routers, gateways, or routing tables and reconfigure them if necessary.

`no cycles found in media db; doing full save`

This informational message is added by **savegrp** to any save set that is saved at the level full instead of the level found in the client's schedule. Due to timing problems, you can occasionally see this message when the clocks on the client and server are out of sync, or when **savegrp** starts before midnight and ends after midnight.

`No 'NSR client' resource for client` *clienthostname*
`savefs: cannot retrieve client resources`

This pair of messages occurs if the client's hostname changed (in */etc/hosts*, NIS or DNS). You may also have deleted the client's "Client" resource while **savegrp** was running. In the former case, you will need to add the client's new name to the aliases attribute of the client (this is a hidden attribute) using **nsradmin** (selecting the "Hidden" display option) or **nwadmin** (selecting the "Details View" option for the Client window). In the latter case, no additional action is required if this deletion was intentional (the next run of **savegrp** will not attempt to save the client). If it was accidental, and you did not want to delete the client, you should add the client back again and add the client back into the appropriate group(s). The next time **savegrp** runs, it will back up the client, just as if the client had been down the previous day.

`no output`

The save set completed, but returned no status output. The most common reasons are that the client crashed or lost its network connection (in other words. a router between the client and server crashed) while the client was being backed up. Another is that the disk on which the client status was being logged filled up (perform a **df /nsr/tmp** to see if this was the case). To determine if the save set was saved, you can use **mminfo**. For example, run **mminfo -v -c *clientname* -t '1 day ago'** and look at the flags column for the completion status. An **a** flag means it aborted. Use a more distant time (the **-t** option) to look further back in time.

filesystem`: No such file or directory`

An explicit save set was named in the Client resource for the specified client, and that save set does not exist (or is not currently mounted) on the client. Make sure you spelled the save set name correctly (and that it is capitalized correctly), and log into the client and verify that the save set is mounted.

n `retries attempted`

`1 retry attempted`

One of these informational messages is prepended to a save set's output if **savegrp** was unable to backup the data on the first try and if the client retries attribute for the group has a value greater than zero. In this case, the specified number of retries was performed before the backup of the save set succeeded or was finally marked as failed.

/path`: This data set is in use and cannot be accessed at this time`

This message is generated by save sets on PC clients running DOS or NetWare. The NetWorker client software on these systems cannot back up files open for writing, due to the interface provided by the operating system. This message actually comes from Novell's TSA and is not changeable.

printer: unknown printer

/*path*/savegrp: printing bootstrap information failed (reproduced below)

/*path*/savegrp: printing bootstrap information failed

These messages, or similar messages, accompany the bootstrap information when **savegrp** was unable to print the savegroup completion notice. You need to either specify a different printer in the printer attribute for the group, or configure your print server to recognize the printer (by default, your system's default printer is used). The bootstrap information is listed as part of the **savegrp** completion mail. You should print out this information immediately, in case your server has a disaster and loses a disk, and fix the printer name used by **savegrp**.

reading log file failed

After the specified save set completed, **savegrp** was unable to read the log file of the output status from the save set. This generally means that someone, or an automated non-NetWorker administrative program or script, removed the log file. This message can also occur if the filesystem on which the client logs are stored has run out of space (use **df /nsr/tmp** to determine if this is the case). Verify that no scripts remove files from */nsr/tmp* (which is where **savegrp** stores the save set log files).

RPC exec on *client* is unavailable. Trying RSH.

This informational message is displayed only when the **-v** flag has been used for verbose information. This message means that **nsrexecd** is not running on the client, and that **savegrp** is attempting to use the **rshd** service instead, for backward compatibility with older versions of **savegrp**.

savegrp: client rcmd(3) problem for command '*command*'

This error message normally accompanies another, more specific, error message. It is generated when the attempt to run the specified command (usually **save** or **savefs** with several command line parameters) failed on the specified save set. The previous line of error output should include the more specific error message (look for that message elsewhere in this section). Generally, the problem is a bad host table configuration, or various permissions-denied errors (server not specified when starting **nsrexecd**, or missing permissions in *.rhosts* if not using **nsrexecd**). If not, log into the NetWorker server as *root* and run the command **savegrp -p -v -c *clientname groupname*** giving the appropriate client for *clientname* and *groupname*. This verbose output should include the necessary additional information needed for fixing the problem.

`Saving server index because server is not in an active group`

This informational message, generated by **savegrp**, means that **savegrp** has noticed that the NetWorker server is not listed in any automatically started, enabled group. Since all the indexes are stored on the server, **savegrp** is saving the server's index and bootstrap information in case a disaster occurs. You should add the server to a group with autostart enabled, or enable one of the groups of which the server is already a member.

`socket: All ports in use`

The NetWorker server has run out of socket descriptors. This means that you have exceeded the socket resource limit on your server. To avoid such future messages, you should determine what other network services are running while **savegrp** is running, and consider rescheduling either **savegrp** or the other services. You can also reduce the parallelism in the nsr_service(4) resource, to reduce the resource utilization.

Errors Messages Generated While Recovering Data

This section lists error messages returned by the NetWorker **recover** program during an on-demand restore using ON-Bar. The messages are organized alphabetically to make them easier to match to the NetWorker message displayed.

`Browsing` *machine*`'s on-line file index`

This informative message explicitly states which NetWorker client's index is being browsed for interactive recovers that resolve to another machine.

`Cannot open recover session with` *server*

Some problem was encountered connecting to the NetWorker server on the named machine.

`error, name is not on client list`

The client invoking the recover command is not in the server's client list. See nsr_service(4) for details.

`Message from server: other clones exist for failed save set`

The **recover** command will be automatically resubmitted to the server, if any files remain to be recovered, because the request failed on a save set that had multiple clones. The server automatically picks a different clone on each attempt.

`Path name is within machine:export-point`

This informative message lets you know that the given pathname is mounted from a network file server and that the recovery will use the index for the named file server. If the machine is not a NetWorker client, then the **-c** option may be necessary.

`/path: Permission denied`

The filename cannot be recovered because you are not root or in the group operator, and you don't have read permission for the file.

`/path: Permission denied(hasacl)`

The filename cannot be recovered because you are not root or in the group operator, the file has an ACL (Access Control List), and you are not the owner of the file.

`Using server as server for client`

This informative message lets you know which NetWorker server was selected for the client's index.

NetWorker XBSA Messages

During a backup or restore, NetWorker attempts to record messages generated by the XBSA library to the file assigned to the NSR_DEBUG_FILE environment variable. If the assigned location is invalid or unreachable, the message is written to one of the following locations:

- the alternate messages directory created during installation, */nsr/applogs*
- the directory assigned to the TMPDIR environment variable
- the */tmp* directory, if TMPDIR is not set

See Appendix A, "XBSA Environment Variables" for descriptions of the NetWorker XBSA variables and values you can assign to them in the **nsrdbmi** script.

NetWorker XBSA error messages appear in the format

```
XBSA-1.0.1 dbmi-1.0 process_id day month date hh:mm:ss year
function_name: BSA_RC_message_code: message
```

`BSA_RC_ABORT_ACTIVE_NOT_FOUND No active object matched the name that was specified for a BSAMarkObjectInactive`

This return code indicates that no active object matching the given search parameters was found in the NetWorker server being used by the NetWorker XBSA session.

`BSA_RC_ABORT_SYSTEM_ERROR System detected error due to` *explanation*`. Operation aborted`

This return code indicates that a general system error has occurred within a NetWorker XBSA function call. This error is returned for all NetWorker errors that do not map cleanly to XBSA errors.

`BSA_RC_APP_OBJECTOWNER_TOO_LONG The appObjectOwner field contained too many characters (`*n* `>=` *n*`)`

This return code indicates that the appObjectOwner field of an ObjectOwner structure contains too many characters and may be corrupt.

`BSA_RC_AUTHENTICATION_ERROR There was an authentication failure for ObjectOwner` *ownername*

This return code indicates that the routine failed to authenticate a BSAObjectOwner with NetWorker server used by the NetWorker XBSA session. The code is returned by the routine BSASetEnvironment to allow for the possibility of changing NetWorker servers during a single session by changing the value of the NSR_SERVER environment option. See Appendix A, “XBSA Environment Variables” for more details on available settings. NetWorker permits all users to back up data and restore their files, without passwords, so this return code should not occur.

`BSA_RC_BAD_CALL_SEQUENCE The sequence of API calls is incorrect. Must call` *item1* `before` *item2*

This return code indicates that an API call sequence was made that does not conform to the *XBSA Data Movement API State Diagram* document.

`BSA_RC_BAD_HANDLE The handle used to associate this call with a previous BSAInit() call is invalid because` *explanation*`.`

This return code indicates that the value passed into the function for bsaHandle contained a NULL pointer.

`BSA_RC_BAD_PARAMETER received parameter` *parm* `with value` *value*`, which is invalid`

This return code indicates that an invalid parameter was received.

`BSA_RC_BSA_OBJECTOWNER_TOO_LONG The bsaObjectOwner field contained too many characters (`*n* `>= `*n*`)`

This return code indicates that the bsaObjectOwner field of an ObjectOwner structure contains too many characters and may be corrupt.

`BSA_RC_BUFFER_TOO_SMALL Buffer is too small to hold the object entry to be returned.` *n* `bytes required for the object entry`

This return code indicates that the buffer is too small to hold the object entry to be returned.

`BSA_RC_COPYGPNAME_TOO_LONG The copyGpName field contained too many characters (`*n* `>= `*n*`)`

This return code indicates that the copyGpName field in one of the supplied structures contained more than BSA_MAX_COPYGPNAME characters and the structure could not be used for the requested operation.

`BSA_RC_DESCRIPTION_TOO_LONG The description field contained too many characters (`*n* `>= `*n*`)`

This return code indicates that the Description field in one of the supplied structures contained more than BSA_MAX_DESC characters and the structure could not be used for the requested operation.

`BSA_RC_INVALID_COPYTYPE the copyType field contained an unrecognized value of` *n*

This return code indicates that the copyType field in one of the supplied structures has a value that is not in the NetWorker XBSA libraries implementation of this enumerated type.

`BSA_RC_INVALID_DATABLOCK the dataBlock parameter contained inconsistent values: bufferLength: n, bufferPtr: n, numBytes: n`

This return code indicates that the fields of a supplied DataBlock parameter are not internally consistent. This may occur when the bufferLength field is less than the numBytes field when data is being sent, or when the bufferLength field is nonzero and the bufferPtr field is NULL.

`BSA_RC_INVALID_KEYWORD an entry in the environment structure is invalid (variable=value)`

This return code indicates that one of the environment strings passed into the function did not have a valid structure. The valid structure of a environment keyword is: KEYWORD=VALUE where KEYWORD is a white space delimited string and VALUE is a white space delimited string followed by a null terminator. This return code can indicate a number of possible errors:

- The KEYWORD string was not in the reserved word list. This error is not returned by the NetWorker XBSA libraries as other environment variables may be passed into the library along with valid keywords.
- The KEYWORD and VALUE strings were not separated by an equal (=) character. This type of error will also be used to detect environment vectors that are not properly terminated with a (char *)NULL entry as well as invalid KEYWORD VALUE pair formats.
- The VALUE string was invalid.
- The VALUE string could not be validated. For example, a hostname string that could not be found by the function gethostbyname().

`BSA_RC_INVALID_OBJECTSTATUS the objectStatus field contained an unrecognized value of n`

This return code indicates that the objectStatus field in one of the supplied structures has a value that is not in the NetWorker XBSA libraries implementation of this enumerated type.

`BSA_RC_INVALID_OBJECTTYPE the objectType is invalid (n)`

This return code indicates that one of the object type parameters, either passed in directly, or contained in an ObjectDescriptor or QueryDescriptor structure, was not in the range of BSAObjectType_ANY to BSAObjectType_DIRECTORY.

`BSA_RC_INVALID_TIME a time field contained an unrecognized value of n`

This return code indicates that an invalid time value was received.

`BSA_RC_INVALID_VERSION the version field contained an unrecognized value of n`

This return code indicates that the version for a parameter passed into the function is not supported by this version of NetWorker XBSA. For routines that receive multiple parameters containing a version field, it does not indicate which parameter is not supported.

`BSA_RC_LGNAME_TOO_LONG The LGName field contained too many characters (n >= n)`

This return code indicates that parameter LGName contained more than BSA_MAX_LGNAME_SIZE characters and may be corrupt. For routines that require multiple LGName parameters, it does not indicate which was invalid.

`BSA_RC_MATCH_EXISTS object matching the specified predicate already exists`

This return code indicates that the object already exists in the NetWorker server being used by the NetWorker XBSA session and the requested operation cannot be completed.

`BSA_RC_MORE_DATA more data is available. Data can be obtained through BSAGetData() or BSAGetNextQueryObject()`

This return code has two meanings in the XBSA Data Movement API:

- *Object Data Retrieval*

  There is more data available for an object being read from the NetWorker server being used by the NetWorker XBSA session. Use BSAGetData to retrieve the next DataBlock from the NetWorker server (see also BSA_RC_BUFFER_TOO_SMALL and BSA_RC_NO_MORE_DATA). This return code is not returned by the BSAGetObjectF function because all data for an object is written to a file descriptor by this function.

- *Query Result Retrieval*

  There are more objects matching the requested query descriptor from the NetWorker server being used by the NetWorker XBSA session. Use BSAGetNextQueryObject to retrieve the next object descriptor from Backup Services (see also BSA_RC_NO_MORE_DATA).

`BSA_RC_NO_MATCH` The *variable* `predicate value of` *value* `does not match the reference value of` *variable*

This return code indicates that no objects matching the specified query descriptor were found in the NetWorker server being used by the NetWorker XBSA session.

`BSA_RC_NO_MORE_DATA there is no more data for the current object`

This return code has two meanings in the XBSA Data Movement API:

- *Object Data Retrieval*

  This return code will be used when all the data for an object being retrieved from a NetWorker server was placed into the given DataBlock parameter for a function call (see also BSA_RC_MORE_DATA).

- *Query Result Retrieval*

  This return code will be used when the last (or only) object matching a query is returned to the caller (see also BSA_RC_MORE_DATA).

`BSA_RC_NULL_APIVERSION an ApiVersion pointer is required`

This return code indicates that a pointer to an ApiVersion structure passed into the function was NULL and is required as input.

`BSA_RC_NULL_BUFFER an buffer pointer is required`

This return code is not used by NetWorker XBSA. A null buffer when reading an object's data (BSAGetData, BSAGetObject) will result in no bytes being read and a BSA_RC_MORE_DATA code begin returned.

`BSA_RC_NULL_DATABLOCK a data block pointer is required`

The DataBlock pointer parameter for the called function was NULL. The caller is responsible for allocating and passing in a DataBlock structure to the NetWorker XBSA library (*see also* BSA_RC_NULL_BUFFER and BSA_RC_INVALID_DATABLOCK).

`BSA_RC_NULL_ENVIRONMENT an environment pointer is required`

This return code is not used by NetWorker XBSA. An environment vector parameter that is NULL will simply not be processed.

`BSA_RC_NULL_NEWTOKEN a value must be entered for the new token. The old token has expired`

This return code indicates that the securityToken parameter newToken was found to be NULL and is required as input. *See also* BSA_RC_NULL_SECURITYTOKEN.

`BSA_RC_NULL_OBJECTDESCRIPTOR an ObjectDescriptor pointer is required`

This return code indicates that the securityToken parameter newToken was found to be NULL and is required as input. *See also* BSA_RC_NULL_SECURITYTOKEN.

`BSA_RC_NULL_OBJECTNAME an object name is required`

The ObjectName parameter passed into the called function was NULL

`BSA_RC_NULL_OBJECTOWNER an ObjectOwner pointer is required`

This return code indicates that a pointer to an ObjectOwner structure was NULL and is required as input.

`BSA_RC_NULL_POINTER a required pointer parameter is NULL`

The NetWorker XBSA library will not return this code. Instead, specific codes indicating that a required parameter was NULL are returned:

```
BSA_RC_NULL_APIVERSION
BSA_RC_NULL_BUFFER
BSA_RC_NULL_COPYGPNAME
BSA_RC_NULL_COPYID
BSA_RC_NULL_DATABLOCK (BSA_RC_NULL_DATABLKPTR)
BSA_RC_NULL_ENVIRONMENT
BSA_RC_NULL_LGNAME
BSA_RC_NULL_NEWTOKEN
BSA_RC_NULL_OBJECTDESCRIPTOR
BSA_RC_NULL_OBJECTNAME
BSA_RC_NULL_OBJECTOWNER
BSA_RC_NULL_OLDTOKEN
BSA_RC_NULL_QUERYDESCRIPTOR
BSA_RC_NULL_RULEID
BSA_RC_NULL_SECURITYTOKEN
BSA_RC_NULL_STREAM
```

`BSA_RC_NULL_SECURITYTOKEN a securityToken pointer is required`

This return code indicates that a pointer to a securityToken parameter is NULL and is required as input. This return code is used internally by the NetWorker XBSA library and should not be seen in normal use. The more specific codes BSA_RC_NULL_NEWTOKEN and BSA_RC_NULL_OLDTOKEN are used as appropriate.

`BSA_RC_OBJECTINFO_TOO_LONG The objectInfo field contained too many characters (`*n* `>=` *n*`)`

The ObjectInfo field passed to the function, either directly or in an ObjectDescriptor data structure, was found to have more than BSA_MAX_OBJINFO characters.

`BSA_RC_OBJECTSPACENAME_TOO_LONG The objectSpaceName field contained too many characters (`*n* `>=` *n*`)`

This return code indicates that the string objectSpaceName contains more than BSA_MAX_OBJECTSPACENAME characters in an ObjectName structure.

`BSA_RC_PATHNAME_TOO_LONG The pathName field contained too many characters (n >= n)`

This return code indicates that the string pathName contains more than BSA_MAX_PATHNAME characters in an ObjectName structure.

`BSA_RC_RESOURCETYPE_TOO_LONG The resourceType field contained too many characters (n >= n)`

This return code indicates that the string resourceType contains more than BSA_MAX_RESOURCETYPE characters and may be corrupt.

`BSA_RC_SECURITYTOKEN_TOO_LONG The securityToken field contained too many characters (n >= n)`

This return code indicates that a securityToken passed in to the function was contained more than BSA_MAX_SECURITYTOKEN characters and may be corrupt. For routines that require multiple tokens, it does not indicate which was invalid.

`BSA_RC_SUCCESS the function was successful`

This return code indicates that the called function did not fail and is returned by all NetWorker XBSA function calls.

`BSA_RC_TRANSACTION_ABORTED the transaction was aborted`

This return code indicates that the current transaction was aborted by the BSAEndTxn function call. A transaction may either be aborted by an internal error or by user request through the Vote parameter to this function.

Glossary

API

An acronym for application programming interface, an agreed-upon set of computer library routines to accomplish a task.

autochanger

A device that has the ability to move media among various components (including slots, media drives, media access ports, and transports) located in the device. Autochangers automate the media loading, labeling, and mounting during backups and recovers.

backup group

A NetWorker client or clients configured to start backing up at a designated time.

blobpage

A physical unit of disk storage used by OnLine Dynamic Server to store *blobspace* data.

blobspace

Large objects, such as multimedia images, are stored in a binary large object space, a logical unit of storage composed of one or more chunks. The physical data is stored in a *blobpage*, and a pointer to this physical location can be stored in a *dbspace*.

bootstrap

At the end of a backup, NetWorker saves the server's media database, NetWorker configuration files, and part of the server index to a special bootstrap save set, which provides vital information for recovering from a disk crash. Save those printouts!

catalog tables

An ON-Bar component that tracks the compatibility of component versions, as well as backup objects and instances.

chunk

A physical unit of disk storage allocated by the system administrator for OnLine Dynamic Server data.

daemon

A program not invoked explicitly, which lies dormant waiting for a specified condition or conditions to occur.

dbobject

Database object, a term which may refer to a *blobspace*, *dbspace*, or *logical log* file.

dbspace

A logical unit of storage that consists of one or more chunks. An OnLine Dynamic Server instance may consist of one *dbspace* or more.

emergency boot file

An ON-Bar ASCII file containing all of the information stored in the ON-Bar catalog tables that pertain to critical dbspaces.

fast recovery

ON-Bar executes a fast recovery by using the physical log to return OnLine Dynamic Server to the most recent point of known physical consistency. Then, the logical logs are used to return OnLine Dynamic Server to logical consistency by rolling forward all committed transactions and rolling back all incomplete transactions.

fileserver

A machine with disks that provide file storage to other machines on a network.

filesystem

1. A sub-tree of a UNIX file tree that is on a specific disk partition or other mount point.
2. A method for storing files.

logical log

A record of OnLine Dynamic Server database transactions stored in a log file. It is used to execute a fast recovery and roll back transactions.

logical unit

A unit of temporary storage that keeps track of where physical units are located.

media manager

The NetWorker component that tracks save sets to backup volumes. The **nsrmmdbd** daemon is responsible for making entries in the NetWorker online media index.

multiplex

A method of transmitting multiple streams of data through the same channel.

mirroring

A storage strategy that pairs a *chunk* of a defined *dbspace* or *blobspace* with an equal sized mirror chunk to enable users uninterrupted access if the primary database fails.

NetWorker client

A machine that can access storage management services from a NetWorker server. Clients may be workstations, PC desktops, or fileservers with gigabytes of data.

NetWorker server

A machine on a network that runs NetWorker software, contains online client indexes, and provides storage management services to clients on a network.

ON-Bar

A backup and recover utility that provides backup and recovery service to OnLine Dynamic Server.

OnLine Dynamic Server

INFORMIX OnLine Dynamic Server, considered by NetWorker as one or more instances of an INFORMIX RDBMS.

online client indexes

Databases on a NetWorker server that contain information pertaining to client backups and backup volumes.

page

A physical unit of disk storage used by OnLine Dynamic Server to read from and write to INFORMIX databases.

physical log

A set of contiguous disk pages where OnLine Dynamic Server stores “before” images of changed pages prior to physically recording the changes.

physical unit

A fixed-sized unit of disk storage allocated for data—chunks and blobpages are examples of physical units.

preconfigured

The initial default selections or configurations for several NetWorker features.

RDBMS

An acronym for Relational Database Management System.

resource

A resource represents a component of the NetWorker software that describes the NetWorker server and its clients. Devices, schedules, clients, groups, and pools are all examples of NetWorker resources. Each resource contains a list of attributes, defining the parameters to use for the configured NetWorker resource. Use the **nwadmin** program to configure NetWorker resources and their attributes.

roll forward (back)

Retroactively applying database transactions from a given point in time into the future (the past) using transaction logs. This is a method for database recovery.

RPC

An acronym for Remote Procedure Call, a protocol that allows a program running on one host to execute code on another host without needing explicitly coded instructions.

save set

A set of files or a filesystem that NetWorker has backed up onto backup media during a backup session. Save sets are assigned a save set ID, an internal number that identifies the backup session for subsequent restoration to primary disk.

shell prompt

Command line prompt, either % or $ (or # for super-user).

storage manager

An application that manages the storage devices and media used for ON-Bar backup and restore requests. Database Module for Informix is a storage manager that connects NetWorker to ON-Bar through XBSA.

XBSA

An acronym for X/Open® Backup Services, which connects NetWorker functionality to Informix ON-Bar. XBSA specifies a programming interface, the XBSA API.

Index

F

G

H

I

L

M

N

O

P

R

S

T

V

W