

# *Microlite* **BackupEDGE**

*Dependable Data Archiving Software*

---

Enhanced  
Data Archiving  
And Recovery  
Software For  
**Open Systems.**

**Technical  
Reference  
Guide**

Read Instructions Before Installing.



Information in this document is subject to change without notice and does not represent a commitment on the part of MICROLITE CORPORATION. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the license agreement.

This document is copyright material and may not be copied or duplicated in any form.

© Copyright 1987-2012 by Microlite Corporation.  
All rights reserved.

The following applies to all contracts and subcontracts governed by the Rights in Technical Data and Computer Software Clause of the United States Department of Defense Federal Acquisition Regulations Supplement.

**RESTRICTED RIGHTS LEGEND: USE, DUPLICATION OR DISCLOSURE BY THE UNITED STATES GOVERNMENT IS SUBJECT TO RESTRICTIONS AS SET FORTH IN SUBDIVISION (C)(1)(II) OF THE RIGHTS AND TECHNICAL DATA AND COMPUTER SOFTWARE CLAUSE AT DFAR 252-227-7013. MICROLITE CORPORATION IS THE CONTRACTOR AND IS LOCATED AT 2315 MILL STREET, ALIQUIPPA PA 15001-2228 USA.**

*BackupEDGE, BackupEDGE SS, RecoverEDGE, Fast File Restore, Instant File Restore, One Touch Restore, BootableBackups and Transparent Media* are trademarks of Microlite Corporation.

All other trademarks, registered trademarks, and copyrights are those of their respective owners.

BackupEDGE Technical Reference Manual - Revision 03.00.03 (ALL)  
January 19, 2012

**Microlite Corporation**

2315 Mill Street  
Aliquippa, PA 15001-2228 USA  
<http://www.microlite.com>  
<ftp://ftp.microlite.com>

(724) 375-6711 - Technical Support  
(724) 375-6908 - Fax  
(888) 732-3343 - Registration Fax  
[support@microlite.com](mailto:support@microlite.com) - EMail

# Table of Contents

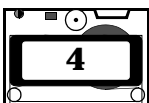
## CONTENTS

1	About This Guide.....	5
2	Terms Used In This Manual.....	6
3	Notational Conventions .....	9
4	EDGE Manual Page.....	11
4.1	Name .....	11
4.2	Synopsis .....	11
4.3	Description.....	11
	Changes from 01.02.xx.....	11
4.4	Arguments and Modifiers.....	12
	Backup related function letters.....	12
	Listing/Verify related function letters.....	14
	Restore related function letters .....	16
	Both Backup and Restore.....	17
	Modifiers used for Backup only .....	23
	Modifiers used for Restore.....	27
4.5	Double Buffering.....	29
4.6	Diagnostics.....	30
4.7	Error Return Codes.....	30
4.8	Compression Performance .....	31
4.9	Backup Summary.....	32
4.10	Restore Summary .....	32
4.11	Level 2 Verify Summary.....	32
4.12	Volume Switching.....	32
4.13	Error Recovery.....	33
	Error Recovery During Backup Operations.....	33
	Error Recovery During Restore Operations .....	33
4.14	Dual Devices.....	34
4.15	Background Operation.....	34
4.16	Signal Processing .....	35
4.17	Screen Input And Output .....	36
4.18	Use Of Wildcards .....	36
4.19	Excluding Files From Being Compressed .....	37
4.20	Specifying Dates And Times .....	38
4.21	Virtual Files.....	39
4.22	Network Backups.....	40

# Table of Contents

---

4.23	Environment Variables .....	41
4.24	Excluding Files From Level 2 Verify .....	44
4.25	Files .....	44
4.26	Limitations .....	45
4.27	Compatibility .....	46
4.28	Special Internal Features .....	46
4.29	Using EDGE With The Menu Environment .....	47
4.30	Standards Conformance .....	47
4.31	Disclaimer .....	47
5	Hints And Useful Tips for /bin/edge .....	49
5.1	Backing Up And Restoring .....	49
5.2	Backing Up Select Files .....	50
5.3	Data Compression .....	51
5.4	Miscellaneous Tips .....	51
	Backing Up From the Root Directory .....	51
	Saving Screen Output in a File .....	52
	Converting Catalog Output to a File List .....	52
	Extra Volume Prompt .....	52
	Negatively Compressed Files .....	53
5.5	Exchanging Tapes With Other Computer Systems .....	53
5.6	Trouble Shooting Common Problems .....	54
5.7	Additional Support Resources .....	59
6	Error Reference Guide .....	61
6.1	Numeric exit codes .....	61
6.2	Fatal Error Messages from /bin/edge .....	61
	FATAL ERRORS WHILE BACKING UP .....	61
	FATAL ERRORS WHILE RESTORING OR VERIFYING .....	63
	FATAL ERRORS GETTING STARTED .....	64
6.3	Warning Messages .....	66
	WARNINGS WHILE BACKING UP .....	66
	WARNINGS WHILE RESTORING OR VERIFYING DATA .....	71
	WARNINGS GETTING STARTED .....	75
6.4	Backup Summary Messages .....	76
6.5	Restore Summary Messages .....	76
6.6	Miscellaneous Error Messages .....	77
6.7	Network Error Messages .....	77
7	Index .....	81



# 1 About This Guide

---

**Microlite BackupEDGE** is a sophisticated data storage and retrieval system for computer systems running a Unix or Linux Operating System<sup>1</sup>. It is designed to provide maximum protection for your data.

*BackupEDGE* can be run in one of three ways...

- 1** as a complete menu driven system that anyone with a basic understanding of the Unix or Linux Operating System can operate, or
- 2** as a completely automated system where nightly backups are performed automatically with no operator intervention, or
- 3** as a standard utility like `tar`, where the user has to have a strong working knowledge of the Unix Operating System.

The manual is intended for those that prefer option **3**. While we *strongly* recommend using the menu system, we recognize that some users must have more control than is offered by it. Other users may find this manual helpful in diagnosing problems or just better understanding how *BackupEDGE* works.

It is assumed that you have already installed *BackupEDGE* from the installation media or from a copy downloaded from the Microlite web site ([www.microlite.com](http://www.microlite.com)).

It is also assumed that you have read and are familiar with the *BackupEDGE* User's Guide.

---

1. The Unix Operating System has many derivatives, including AIX, LINUX, SOLARIS, etc. For simplicity, the word Unix will be used throughout this guide.

## 2 Terms Used In This Manual

---

**Absolute Pathname:** A filename beginning with a slash (/). A file saved with an absolute pathname (such as `/etc/termcap`) may only be restored to the `/etc` directory.

**Archive Device:** The floppy disk drive, tape drive, or cartridge drive used to backup and restore files. You may also save and restore to a regular file.

**Archive Media:** The diskette, reel or cartridge tape, or disk cartridge used to store your data. Please note that floppy disks, disk cartridges, and some cartridge tapes must be formatted before use with *BackupEDGE*. See your operating system manual for more details.

**Auto Changer:** A device containing one or more tape drives and one or more tape storage slots. Tapes may be moved automatically between storage slots and tape drives by *EDGE*. Also known as a *Library* or *Autoloader*.

**Autoloader:** See *Auto Changer*.

**Background, Foreground:** Background and foreground have special meaning to the Unix Operating System. Foreground tasks are generally run in interactive mode, meaning that information from the program is displayed on the screen and input is typed on the keyboard. Background tasks run as “unattached” programs requiring no display output or keyboard input. They can be started automatically by the Unix `cron` or `at` scheduling programs, or by a foreground program.

**Binary File:** A file containing characters other than those in the ASCII decimal range of 32 to 127 (hex 20 to 7f). A compiled C program is an example of a binary file.

**Blocking:** Unit of measure. typically 512 characters, or bytes.

**Blocking Factor:** The number of 512 character segments of data that can be read or written at one time.

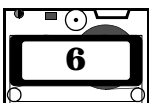
**Cron:** A program that always runs when the operating system is in multi-user mode. It constantly looks in a set of files called `crontab` files for programs to run and times to run them.

**Device:** A device is a piece of hardware, such as a disk drive or a printer, that is attached to a computer. Every device is assigned a name, or *Device Node*.

**Device Node:** The name which the operating system uses to access a physical device. For example, `/dev/hd0` is the one possible name for a primary hard disk, while `/dev/lp0` is a typical name used to access a line printer. Devices are found in the `/dev` directory.

**File System:** A file system is a hierarchy of files and directories mounted under the root directory.

**Differential Backup:** A backup of any files or directories that have been created or modified since the last **Master Backup**. Historically, these were called *Incremental Backups* by *BackupEDGE*.



**Level 1 Verify:** A method of reading back an archive and checking for media readability and file header integrity.

**Level 2 Verify:** A method of reading back an archive and comparing each file on a character by character basis against the actual file on the hard disk. Also known as a **Bit-Level Verify**.

**Library:** See **Auto Changer**.

**Link:** The technical term for a filename. A single real file may have more than one link, or filename.

**Master Backup:** A full backup of the complete system, including any mounted file systems.

**Raw Filesystem Partition:** A disk partition managed by an application program instead of a Unix filesystem. Applications such as Oracle, Informix and Sybase sometimes store their data in raw partitions.

**NOTE:** *BackupEDGE* can properly archive and restore **Raw Filesystem Partitions**, but they **MUST** be identified as raw in advance. See “Raw Filesystem Partition Special Support (master.cfg)” on page 65 for more information and instructions on identifying virtual files.

**Regular File:** A standard file whose actual size (in bytes) is always reported by the operating system correctly. Regular files may contain programs or data.

**Relative Pathname:** A filename beginning with a dot (.). A file saved with a relative pathname (such as `./etc/termcap`) will be restored relative to the current working directory at the time of the restore.

**Resource:** A named set of properties describing one device for *BackupEDGE*. For example, a resource may represent a tape drive, and include the appropriate device node(s) for it, the hardware block factor, default *BackupEDGE* block size, and so on. Resources are created / modified with the **EDGE.RESMGR** program.

**Shell:** The Unix Operating System command interpreter, normally run when the user logs in. Typified by the prompt commands `%`, `$`, or `#`.

**Seeking Device:** A device capable of moving its read / write head directly and randomly to any spot on the media.

**Shell Program or Shell Script:** A series of shell commands run sequentially from a file list.

**Symbolic Link:** A special link type. The file contains the path to a file located elsewhere on the system or across a network.

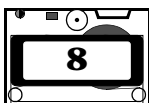
**Tar:** tape archiver, A backup utility program.

**Virtual File:** A special kind of file used by some application programs. The filesystem interprets this file type as being quite large, while a special technique is employed to

prevent currently empty sections of the file from consuming actual disk space. Virtual files cannot be copied, archived or restored with standard operating system commands.

**NOTE:** *BackupEDGE* can properly archive and restore **Virtual Files**, but they **MUST** be identified as virtual in advance. See “Virtual File Identification (master.cfg)” on page 65 for more information and instructions on identifying virtual files.

**Volume:** One disk, tape, cartridge, etc. from a full backup set.

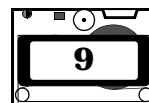


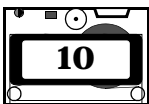
### 3 Notational Conventions

---

The following conventions will be used throughout this manual.

- **Typewriter font** will be used for commands that you must type, for computer generated prompts, and for references to file names.
- **[Key]** will be used to indicate single keystrokes. For example, **[Enter]** means to press the **Enter** key. Do not type in the word **ENTER**. If two keys appear in sequence, for example, **[Ctrl-A]**, press and hold the first key, **Ctrl** (or **Control**), press the second key, **A**, then release both keys simultaneously.
- **Select** means to use one of the four **arrow** keys to position the cursor over the menu item desired, then press **[Enter]**.
- **Choose** means to navigate to a menu or program by going through a series of submenus. For example:  
`EDGEMENU -> Schedule -> Nightly Scheduling` means to run the `edgemenu` program, then select the `Scheduling` menu, and select the `Nightly Scheduling` option by using the **arrows** and **Enter** key.
- Full screens are used to represent the actual displays you will see as you run the programs. **Bold** and **BoldItalic** are used to highlight special items of interest.
- A program name will generally be referenced in bold caps, even though you must type the program name in lower case to execute it. For instance, **EDGEMENU** would refer to the main menu program `/usr/bin/edgemenu`, **EDGE** would refer to `/bin/edge`. **EDGE.TAPE** would refer to `/usr/lib/edge/bin/edge.tape`, etc.





# 4 EDGE Manual Page

---

## 4.1 Name

---

edge - Tape or Floppy Archiver (Version 03.00.03)

## 4.2 Synopsis

---

```
edge -{MdcCpTtXnU} [vVfbehnkllLpmGEFZSdR0-99ASw] [tape] [block  
size] [compression limit] [archive size] [-zKEYWORD] file1  
file2...
```

## 4.3 Description

---

**EDGE** saves and restores files on magnetic tape, floppy disk, CD/DVD, FTP site, attached storage, or add-on hard disk. **EDGE**'s actions are controlled by a key argument. The key is a string of characters containing one or more function letters and possibly one or more function modifiers. Other arguments to the command are file or directory names specifying which files are to be archived or restored. A directory name refers to the files and to the recursive subdirectories of that directory.

**NOTE:** Any backup performed outside of EDGEMENU or a Scheduled Job will not use any Backup Domain or Backup Sequence information to record it. It is recommended that you do use EDGEMENU or a Scheduled Job to perform backups instead. This procedure is detailed in the Integrations Guide, contained in this manual.

**EDGE** permits a file to extend across media boundaries. **EDGE** can archive and restore an entire file system. All files and directories will be restored. This includes empty directories, special character and block device files, named pipe files, linked files, and symbolically linked files. **EDGE** can handle dual drives of differing storage capacity in a sequential manner with automatic switching between drives. Additionally, **EDGE** has built in many error recovery capabilities should the backup media develop a fault.

### Changes from 02.00.xx

You may specify a resource name with the **f** flag, rather than just a device node. If you do this, you do not need to specify **k** or **b** options. These will be taken from the resource. Likewise, software compression settings from the resource will be used as well. The resource can be any resource that could be used with EDGEMENU, including a tape, CD, DVD, URL, FSP, etc.

If using a URL or FSP, be sure to see the **-zSLOTNAME** option below.

## **Changes from 01.02.xx**

The **I** flag previously performed a Differential backup. Now, this flag produces a usage message. To perform a Differential backup, use the **D** flag. In previous versions, the **D** flag was used to back up empty directories, but this has been the default behavior for quite some time. Effectively, the **D** flag has done nothing. Please note that using the **D** flag with the **M** flag, which was a common practice in 01.02.xx now produces an error message designed to catch legacy uses of the **D** flag.

The **Allmounts**, **Netmounts**, and **Readmounts** special filesystem excludes now do not make an exception for directly included directories. For example,

```
edge cvf /dev/null -E Allmounts /mounted_directory
```

will back up only the directory entry itself, not its contents (assuming, of course that `/mounted_directory` is a filesystem mount point). This is more consistent with normal exclude processing. Also note that `/` is not excluded by **Allmounts**.

The format of the summary provided by **EDGE** has been changed to be more consistent with the summaries generated by a *Scheduled Job* or *EDGEMENU* backup.

When computing the data transfer rate of a backup including software compression, **EDGE** now uses the amount of data written to tape rather than the amount of data encountered, as the basis of the calculation. In other words, it now measures the speed of your archive device.

The **EDGE** archive format has been improved to include:

- Virtually unlimited pathname / symlink lengths
- ISO 1003.1-2001 compatibility (**tar** / **pax**)
- Streaming compression (no intermediate storage during backup)
- Single-pass virtual file backups
- A label at the front of every volume
- Data checksums in addition to header checksums
- A reliable end-of-archive marker, for better multi-volume handling

Of course, **EDGE** can still read older **EDGE** archives.

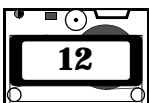
## **4.4 Arguments and Modifiers**

---

### **Backup related function letters**

The function portion of the key argument is specified by one of the following letters...

- c** creates a new archive. Writing begins at the beginning of the archive media rather than the last file. All previous data is erased. If no files are specified, then the previous data will not be erased. If the **n** option is set (seeking devices) or the **P**



option is set (non-seeking devices) then data will be compressed if the file size is over 4 blocks or the size given by the **L** option. Executable files will not be compressed unless the **Z** option is included as a modifier. Files with names that end in **.gz**, **.bz**, **.bz2**, and **.tgz** are assumed to have been already compressed by another utility. No attempt at compression will be made for these files.

**C** Identical to **c** above except that compression is turned off.

**M** This is the **Master Backup** option. **EDGE** will do a complete backup of the system including all mounted file systems. This backup should be done from the root directory. If the **Master Backup** completes successfully, the date and time of the start of the backup will be put in the file **./etc/Master\_backup** which is available for viewing. A sample invocation would be...

```
cd /  
edge Mvbkf 64 149760 /dev/tape .
```

If there is no **./etc** directory present **EDGE** will put the date in the file **./Master\_backup**. The date is updated only if the **Master Backup** completes successfully and is not interrupted. During the **Master Backup**, **EDGE** will place an unenforced lock on each file as it is backed up. The lock will prevent writing but allow reading. See the information on the locking option modifiers below for more information. The **M** option automatically invokes the **D** modifier to backup directory permissions and the **l** modifier to report unresolved links. If the device library indicates that the device can seek, data files over 15 blocks (or the limit specified by the **L** modifier) are compressed.

**D** This is the **Differential Backup** option. Files modified since the last **Master Backup** will be archived. **EDGE** will traverse the entire system including all mounted file systems and look for any files that have been added or changed since the last **Master Backup**. Two different methods of detecting added or changed files can be used, resulting in a **Level 1** or a **Level 2 Differential Backup**. See the descriptions below. **Level 2** is the default. The date of the last **Master Backup** is taken from the last modification time of the file **./etc/Master\_backup**. Do not edit or modify this file, as **EDGE** uses the creation time of the file and not the text contents to determine the **Master Backup** time. This option should be invoked from the root directory. Example...

```
cd /  
edge Dvbkf 64 149760 /dev/tape .
```

The **Differential Backup** option automatically invokes the **D** modifier to backup directory permissions.

**Level 1 Differential Backup.** This compares the file modification time (*mtime*) of each file against the start time of the last successful **Master Backup**. If *mtime* for the file is newer, then the file is archived. Modifications to a file which change *mtime* include creation, writing, and updating.

**Level 2 Differential Backup.** This compares the file change time (*ctime*) of each file against the start time of the last successful **Master Backup**. If *ctime* for

the file is newer, then the file is archived. Modifications to a file which change **ctime** include creation, writing, updating, moving, linking, and changing mode or ownership.

Some programs (like `tar` and `cpio`) purposely modify **mtime** when they restore a file. Therefore if a new program is installed, or if a file is restored from a backup, its **mtime** may be set to a time previous to the last **Master Backup**, which means that a **Level 1 Differential Backup** will ignore it. So a **Level 2 Differential Backup** (which is the default) is much more robust, although it may take up more archive space.

Please note that this option was `'-I'` in previous versions. There is no way to perform an **Incremental Backup** with `/bin/edge` except by using the `-zDATE` option. Use Scheduled Jobs instead for this.

- P** Pack-regardless option. This option will allow compression of files to a non-seeking device such as a tape. Use this option when tape backups start using more than one tape, or if you wish to save network bandwidth. The **P** option implies the **c** option. For example...

```
edge CPZvbkf 64 149760 /dev/tape /
```

will back up the entire file system compressing all files over 4 blocks to a 150 Megabyte tape.

Unlike previous versions, this option does not require temporary hard disk space in order to compress files.

---

## Listing/Verify related function letters

- t** Lists the names of all files on the archive. If used without the **v** modifier only the filenames are listed, otherwise the filenames as well as permissions and modification times are listed. If the device can seek, then a rapid listing will occur. An internal default to a block factor of 1 is used in this case. The following will list all files on the tape device `/dev/tape`.

```
edge tvbkf 64 149760 /dev/tape
```

If you want to see if a few particular files reside on the archive, you may put the filenames you are interested in on the command line. If those files are present on the archive they will be listed. For example,

```
edge tvbkf 64 149760 /dev/tape /usr/rworld/index00
```

The wildcards `"**"` and `"?"` may be used. If wildcards are used on an **EDGE** command line, they must be enclosed in single or double quotation marks. Only those files with names matching those on the command line will be displayed. For example, the following command would list all files on the archive with names ending in `.dat`.

```
edge tvbkf 64 149760 /dev/tape "**.dat"
```

- T** This is the "Table of contents with verification", also known as **LEVEL 1 Data Verification**. All the data on the archive will be read into memory. The file headers contain a checksum and these will be recalculated and compared to the

original. The archive media will be checked to make sure that every byte of data is readable. The file sizes given by the file description headers are compared to the actual byte spacing between headers on the media. If the archive was made with data checksums (see the `-zCHKSUM` option), then those checksums will be recomputed and compared against the archive. If the archive contains any encrypted files, then you will be prompted for the passphrase.

This level does *not* do any comparison of the file to that found on the file system hard disk. There are two possible errors. The first is a directory checksum error. This means that the checksum of the header or data (if the archive includes data checksums) does not match the recalculated checksum. This would appear as...

```
edge: Tape Write ERROR 5 occurred
```

This means that the media is corrupt. On encountering this error, **EDGE** will switch into ERROR RECOVERY mode and attempt to keep going. This allows you to see how bad your archive is and how much of it is really recoverable.

**TT** This is the “Table of contents with bit level verification”, also known as **LEVEL 2 Data Verification**. **LEVEL 2 Verification** encompasses a **LEVEL 1 Verification** with the addition of a comparison of the files on the archive media to those on the file system hard disk. For example the command...

```
edge TTfbk /dev/rct0 64 149760
```

would perform a **LEVEL 2 Verification** on the media of the tape device with a block factor of 64 and tape capacity of 149760 Kilobytes (150MB).

**EDGE** will first check to see that the size of the file on the archive media is that same as that on the hard disk. If the sizes are different, the message `<!Size>` is printed next to the file name. This signifies that the file sizes do not match so there is no point in doing a byte by byte file compare. The output of **EDGE** would be as follows...

```
./usr/adm/messages, 12 blocks <!Size>
```

If the sizes and match, then a byte by byte file comparison is done. If there is a perfect match for every byte in the file, the message `<V>` is printed beside the file name so that the user is aware that there is no discrepancy. If there is a mismatch, then **EDGE** has to determine if the file on the hard disk has been modified since the backup. If the time stamp has changed, then it prints the message:

```
./usr/adm/messages, 12 blocks <!Date>
```

to notify the user of this fact. If the file on the hard disk does not appear to have been modified since the backup, then the message `<!Byte>` is printed telling the user that there is a byte mismatch between the file on the archive media and the file on the file system. For example the output...

```
./etc/passwd, 2 blocks <!Byte>  
./etc/group, 2 blocks <V>
```

would signify that the `/etc/group` file was unchanged and verified properly, while the `/etc/passwd` file does not match the archive, but also did not appear to have been changed since the backup was done.

The difference between `<!Date>` and `<!Byte>` is significant: `<!Date>` indicates that some other program has modified the file, and thus it is okay that it does not match the archive data. `<!Byte>`, on the other hand, indicates that the file seems to be untouched, yet the archive does not contain the right data for it. In this case, the archive is probably corrupt. This is a serious error!

If any file shows a difference such as `<!Date>` or `<!Byte>`, then there is a mismatch. The details are recorded in a separate file referred to as the **CHANGEFILE**. This file contains the details of only those files that don't verify exactly. This file can be used to review the status of the **EDGE** verification, to see if any differences detected are important.

If the files had a mismatch in size, the size of the file on the archive media is given along with that of the file on the hard disk file system. If the files had a mismatch in data, the block number of where the mismatch occurred is given. The block offset is given in tape blocks which are 512 byte blocks.

If the file on the hard disk cannot be accessed because of inadequate permissions or it has been removed, this fact is also recorded. The message...

```
"<!Not checked>"
```

is given in the screen output. The **CHANGEFILE** will contain the reason that this file was not checked.

If a file is in compressed format on the archive media, it is expanded in memory and compared to the file on the file system hard disk. No intermediate files are created by this process and the expansion takes place in conjunction with the comparison so that one does not run out of memory.

The reason for re-expanding the file on the archive media is to guarantee that it can be expanded properly (no corruption in the original compression process) and that the final output is identical to the original file. This gives an additional level of confidence when dealing with compressed files.

**LEVEL 2 Verification** will produce a summary when completed. The summary includes the number of files that were different and the number of files that could not be checked because of inadequate permissions or because they are no longer present.

For command line operations, the **CHANGEFILE** file name is set to `/tmp/ChangedFiles`. The menu system checks `/etc/defaults/edge.cfg`, which by default sets it to `/usr/lib/edge/lists/ChangedFiles`.

---

## Restore related function letters

- x Extracts (restores) files from the archive media. If no file arguments are given, the entire content of the archive is extracted. Pathname arguments may be used. Arguments may include the wildcard characters `"**"` and `"?"`. Arguments containing wildcards should always be quoted on the command line. If the argument is a directory present on the archive, then the entire directory is recursively extracted. If multiple versions of the same file are on the tape, the last

version will overwrite all preceding versions. If a file is in compressed format, it is decompressed in a streaming fashion directly to the hard disk without any intermediate files. This prevents disk fragmentation.

- N** Non-destructive restore. This is the same as the **x** option except that, if a file is already present, it will not be overwritten. This is particularly useful for upgrading to new versions of operating systems.

It is also quite useful if a set of files has been wiped out but you are not sure exactly which files have been affected. A non-destructive restore will bring back any file not present, but will not affect files already present on the hard drive(s).

Keep in mind that if you are restoring a **Master Backup** and a **Differential Backup** using the non-destructive restore feature, you must first restore the **Differential Backup**, then the **Master Backup**.

Device files, such as are found in `/dev`, are always restored in this way, unless the `-zREPLACE_ALL` flag is given..

- U** Update restore option. All files on the disk that are older than those on the archive are updated. This ensures that only the newest copy of a file is restored. If a file does not exist on the hard drive(s) it will be created.
- q** Restore only the first file on the archive, then exits. Useful for reading or restoring tape labels. Archive must be in a `tar` or **EDGE** compatible format for this to work.

**NOTE:** The following characters modify the above key function letters...

---

## Both Backup and Restore

- v** Verbose option. Causes **EDGE** to give more information about the backup, the listing, or the restore operation in progress. When used with the **t** function, **v** gives more information about the entries than just the name. Without the **v** option, **EDGE** will do its work silently unless there are error diagnostics.
- V** This is identical to **v** above except that a catalog of the files backed up, verified, listed, or restored is created. This file is a carbon copy of the screen output from **EDGE** and includes error and warning messages. The name of the catalog file is...

`/usr/lib/edge/lists/{prefix}_{month}.{day}`

The term prefix above is symbolic for one of the following terms depending on context...

**Backup Master Increm Verify Listing Restore**

For example, if the backup was made on June 17th, the catalog filename would be...

`/usr/lib/edge/lists/Backup_Jun.17`

Further examples are shown below...

Regular Backup (C/c)	<code>/usr/lib/edge/lists/Backup_Jun.17</code>
Master Backup (M)	<code>/usr/lib/edge/lists/Master_Jun.17</code>
Differ. Backup (D)	<code>/usr/lib/edge/lists/Differ_Jun.17</code>

Verify (T) or (TT)	<code>/usr/lib/edge/lists/Verify_Jun.17</code>
Restore (x)	<code>/usr/lib/edge/lists/Restore_Jun.17</code>
List Files (t)	<code>/usr/lib/edge/lists/Listing_Jun.17</code>

If the default directory `/usr/lib/edge/lists` is not present, the catalog file is placed in the current directory. If the environment variable `EDGECHAT` is set to a valid directory name, this name will be used as the directory in which the catalog will be stored. You must be sure that the environment variable is exported as follows before invoking **EDGE**...

```
EDGECHAT=/tmp
export EDGECHAT
```

In the above example, the catalog files will go into the `/tmp` directory.

For user convenience in keeping up with archives, the file `LAST_{prefix}` is automatically linked to the last catalog file made of it's type. Thus the file `LAST_Master` will always be the catalog of the last **Master Backup** made. The file `LAST_Increm` will contain the catalog of the last **Differential Backup**. It is not called `LAST_Differ` for historical reasons.

After a period of time, the catalog directory can get rather cluttered. This can be cleaned up by removing all files except those starting with `LAST_`.

If the user wants the catalog to go to a specific file other than the default names **EDGE** assigns, the name of the file can be specified using the environment variable `EDGEFILE`. For example, the following will allow the catalog file to be placed in the file `/usr/lib/edge/lists/CATALOG`:

```
EDGEFILE=CATALOG
export EDGEFILE
```

If the environment variable `EDGECHAT` were set to `/tmp` as above, then the catalog file would be saved in `/tmp/CATALOG`.

If the file specified in `EDGEFILE` contains a `/`, then it is assumed to be an absolute or relative pathname, and `EDGECHAT` is ignored.

If `EDGEFILE` is set to the keyword `NONE`, no catalog file will be made except for `LAST_{prefix}`. This file is placed in the directory specified by the shell variable `EDGECHAT`, or the default directory if this variable is not set.

- b Causes **EDGE** to use the next argument as the blocking factor for archive records. The default is dependent on the version of Unix, but is typically 30. For improved performance, a block factor of 6 to 36 is recommended for floppy diskettes. 64, 128 and larger are recommended for tapes, typically some even divisor of the tape drive's internal buffer size. The maximum is implementation specific and usually dependent on the operating system and available memory. Typically, this is 127 to 4096000 blocks. In practice, the block size should always be an even divisor of the volume size (`k` or `s`).

```
edge cvbkf 64 149760 /dev/tape .
```

If you choose a high blocking factor (above 64), it is **CRITICAL** that you perform a successful attended **Master Backup** and **Level 2 Verification** before using the

settings in production. Some tape drives may lose data when used with large block factors, and a **Level 2 Verification** is the only way to reliably detect this.

If you have specified a resource name with the **f** option rather than a device node, then **b** is not needed. Using it will override the resource setting.

- f** Causes **EDGE** to use the next argument as the name of the archive or resource rather than the default of `/dev/edge`. When this option is used, **EDGE** assumes the device is non-seeking unless overridden by the **n** option.

```
edge cvbkf 64 149760 /dev/tape .
```

If you are specifying a device node, be sure to include a `/` in the name so that **EDGE** realizes that it is not a resource name. You may instead specify a resource name, such as `tape0`, to use the settings in the Resource Manager for this resource. You may write to any type of resource that you could with **EDGEMENU**, including CD/DVD resources, URLs, FSPs, etc. Please note that the `-zSLOTNAME` option lets you specify a slot name for those resources that use it.

If the name of the file is dash (`-`), **EDGE** writes to the standard output or reads from standard input. Thus you can move hierarchies with the command...

```
cd fromdir; edge cf - |(cd todir; edge xf -)
```

**EDGE** can also use remote tape devices, simply by including a system name with the device name:

```
edge cvbkf 64 149760 mlitedom:/dev/tape .
```

See the sections on **Network Backup** and on the `REMOTE_PUT` and `REMOTE_GET` environment variables for more information.

- k** **EDGE** reads the next argument as the size of the volume in kilobytes. Very large files are split into extents across volumes.

```
edge cvbkf 64 149760 /dev/tape .
```

When used with the **v** option, the screen output will include the room left on the current volume as...

```
VOL=XXXXU
```

where `XXXX` is the room left in Units, which may be kilobytes (KB), Megabytes (MB) or Gigabytes (GB). This enables the user to determine when the current volume will need to be changed.

If you specify a resource name with the **f** option, **k** is generally not needed. You may override the resource setting with it if you like.

- g** Group option (see **G** below).
- G** Group option. **EDGE** will change the group id of all files being copied or extracted to the next argument. This is very useful for installing a set of files for later removal. No matter where the files go on the hard disk, they will carry with them the group id given. Later they can be removed with the `find` utility as follows...

```
find / -group XX -exec rm -i {} \;
```

- n The device is not a magnetic tape and thus can seek. Unless the **P** option is used, compression will only function with seeking archive devices. In general, any time the **f** modifier is used, the device is considered to be non-seeking and compression is turned off. When the **n** modifier is used along with the **f** modifier, it indicates that the device is capable of seeking. The **cnf** option string is typically used to force compression on seeking devices. The **tnf** option will enable seeking to force a rapid listing of the archive.
- F File option. **EDGE** will get a list of file names to use for archiving or restoring from the file name specified in the next argument. Any number of files can be in the list, each on a separate line. This option works for both backing up and restoring files. Additionally, if the name of the file is a dash (-), the standard input is used to obtain a list of files to backup or restore.

For example, the following command will back up all source code files (any file ending in **.c**) that have changed within the last seven days...

```
find / -name "*.c" -mtime -7 -print | \
edge cvbkfF 64 149760 /dev/tape -
```

Filenames that do not start with **.**, **...**, or **/** are treated as position-independent patterns, rather than normal filenames or patterns. A normal pattern is required to match from the start of the filename. For example, **./usr/lib/edge** will match only those files which start with **./usr/lib/edge**. A position independent pattern, such as **edge** will match any filename that contains a component called **edge**. It will therefore match **./usr/lib/edge/bin**, **./usr/lib/edge**, **./edge**, etc. It will not match **./usr/lib/edge\_directory**. To match this, one might use the position independent pattern **edge\***.

Position-independent patterns are found by traversing the current directory, and are backed up in relative format (i.e., starting with a leading **'.**).

- E Exclude option. Used to exclude files or directories from being archived or restored. For example...

```
edge cvbkfEE 64 149760 /dev/tape /usr /dev .
```

would exclude two directories (**/usr** and **/dev**), while

```
edge cvbkfEE 64 149760 /dev /usr/adm/messages .
```

would exclude one directory and one file.

When used to exclude during a restore, the spelling must match exactly that used on the archive. The leading **"/** or **./** does not have to match exactly.

During a restore, if the filename matches exactly what is on the archive, then it is excluded. However, it is also excluded if the filename requested would match the one on the archive, if both had **'.'** replaced with the working directory that was used when the archive was made, and **'..'** removed.

For example, the following would restore one file:

```
cd /usr
edge cvf /tmp/test.edge .
```

```
cd /any_directory_at_all
edge xvf /tmp/test.edge /usr/lib/edge/bin/edge
```

This file would restore “./lib/edge/bin/edge” relative to “/any\_directory\_at\_all”, since it was stored on the archive with a leading “./” while the working directory was /usr. If it had been archived with:

```
edge cvf /tmp/test.edge /usr
```

then the restore command would replace the original file.

One nice feature of this is that excluding “/proc” on a restore

The wildcard characters “\*” and “?” may be used for filename pattern matching. Any wildcard expressions should be quoted with either single or double quotes when used on a command line. For example, the following would exclude all index files (those ending in .idx) from a **Master Backup**...

```
cd / edge MvbkfE 64 149760 /dev/tape `*.idx` .
```

The **E** option using wildcards also applies to directories. For example, if you want to exclude all the **Wastebasket** directories, you could use the following command...

```
edge cvbkfE 64 149760 /dev/tape `*/Wastebasket` .
```

The **E** option can be used to exclude all Read-only mounted file systems using...

```
edge cvbkfE 64 149760 /dev/tape Readmounts .
```

The **E** option can be used to exclude all NFS mounted file systems using...

```
edge cvbkfE 64 149760 /dev/tape Netmounts .
```

**EDGE** looks at the mount command to determine which filesystems are NFS mounted. Mount entries that start with a `node_name`, followed by a colon (:), followed by a directory name, are considered NFS mounted.

Additionally, the **E** option can be used to exclude all mounted file systems (except root) regardless of mount points as follows...

```
edge cvbkfE 64 149760 /dev/tape Allmounts .
```

This option is helpful when portions of the operating system must be archived while other users are logged in.

Be careful when using these arguments. For instance, many systems now have a separate /**stand** filesystem containing the kernel and boot programs. In many cases, this filesystem is also always mounted read-only.

Using the **Readmounts** or **Allmounts** flags to back up the root filesystem will also exclude these critical files from being archived. In this instance it would be better to explicitly exclude all filesystems except root and /**stand**.

- x File Exclude option. Similar to **E** option, but with the **x** option, **EDGE** will get a list of file names to exclude from the file name specified in the next argument.

Like the **E** modifier, you can use wildcards in the list. You may NOT use the keywords **Readmounts**, **Netmounts**, or **Allmounts**, as these could represent actual filenames you wish to exclude.

0-99 These numbers refer to an entry in the **Device Library** of available devices. The library can have up to 100 entries (0-99), and specifies the various parameters associated with a archive backup device. For example, to do a **Master Backup** to device 7 in the library use...

```
edge Mv7 .
```

The file used for the **Device Library** is /etc/default/edge. If this file is not available then /etc/default/tar is used. The **Device Library** is an ASCII file that can be edited with a simple editor. Sample lines in the **Device Library** look like this...

```
#  @(#) edgedefsco.src 1.4 06-10-1998
#  device                block  size    tape
archive0=/dev/rfd048ds9  18    360    n
archive1=/dev/rfd148ds9  18    360    n
archive2=/dev/rfd096ds15 30    1200   n
archive3=/dev/rfd196ds15 30    1200   n
archive4=/dev/rfd0135ds9 18    720    n
archive5=/dev/rfd1135ds9 18    720    n
archive6=/dev/rfd0135ds18 36    1440   n
archive7=/dev/rfd1135ds18 36    1440   y
archive8=/dev/rctmini    64    37632  y
archive9=/dev/rctmini    64    79104  y
archive10=/dev/rctmini   64    119040 y
archive11=/dev/rct0      64    45312  y
archive12=/dev/rct0      64    59904  y
archive13=/dev/rct0      64    124416 y
archive14=/dev/rct0      64    149760 y
archive15=/dev/rct0      64    249600 y
archive16=/dev/rStp0     64    319488 y
archive17=/dev/rStp0     64    524544 y
archive18=/dev/rStp0     64    999936 y
archive19=/dev/rStp0     64    2522880 y
archive20=/dev/rStp0     64    1211888 y
archive21=/dev/rStp0     64    1299456 y
archive22=/dev/rStp0     64    1951488 y
archive23=/dev/rStp0     64    3972096 y
archive24=/dev/rStp0     64    11800329 y
archive25=/dev/rStp0     64    9999360 y
```

The format consists of four fields separated by tabs or spaces. The first field specifies the name of the device and must be preceded by the word **archiveX=** where **x** is a number between 0 and 99. The second entry is the block size in tape blocks (512 bytes) to use for the device. The third entry is the device capacity in kilobytes. The fourth entry is a either a **y** or **n** depending on whether the device is a tape or not.

When a digit 0-99 is used to represent a device in an **EDGE** command line, the **f**, **k**, **n**, and **b** options are never needed.

**-zDATE={mm/dd/yyyy-hh:mm}**

Backup or restore all files modified on or after the specified date. This option allows the user to use a specific date and time to backup or restore a set of files. The time is

specified in military (24 hour) time; if no time is given, midnight is assumed. See “Specifying Dates And Times” for more information on date and time specification. For example, to back up all files modified on or after June 1, 1998 at 6:30PM the following command would be used...

```
edge cvbkf 64 149760 /dev/tape -zDATE=6/1/1998-18:30 .
```

This option can also be used for restoring files based upon the date of last modification. For example, to restore files from a **Master Backup**, but only those modified on or after June 15, 1998 at midnight, the following command would be used...

```
edge xvbkf 64 149760 /dev/tape -zDATE=6/15/1998
```

#### **-zSWAB**

This byte-swaps all data. If this option is used when backing up data, the data is byte-swapped just before being written to the archive device. On restore, the data is byte-swapped just after being read from the device. This option is designed to allow easy transfer of data between systems with different CPU architectures.

#### **-zBG**

Background Mode. This forces **EDGE** to behave as if it was launched in the background. Error codes match those issued as background error codes, and prompts are answered with built-in defaults.

#### **-zSLOTNAME**

This overrides the default slot name of `default` with the name specified, such as `-zSLOTNAME=mybackup`. When backing up to a resource that allows multiple archives, specifying a slot name is a good idea. Remember that a new backup will overwrite any existing backup with the same slot name, if the medium allows it. When reading an archive, this selects the named archive automatically instead of prompting, if the medium contains more than one archive.

---

## **Modifiers used for Backup only**

- s** Speed option. Initiates **Double Buffering** using shared memory, semaphores and dual processes. In some instances, this will significantly speed up the backup process. Five shared memory buffers, each the size of the blocking factor (specified with the **b** option) are created to facilitate data transfer. More buffers may be allocated with either the **ZBUFFERS** environment variable or the `-zBUFFERS=` command line argument. See “Double Buffering” for more details.
- L** Specifies the limit (default of 4 512-byte blocks) used for compressing files. Any file larger than this limit will be compressed.
- z** This makes compression include executable files. This option implies **c**.
- l** Tells **EDGE** to notify you if it cannot resolve all the links to the files being backed up. This is the default for **Master Backups**. A list of unresolved linked files is printed at the end of the backup.

e Reserved. Previous versions of **EDGE** used this to keep files from being split across volumes.

h Archive the contents of the symbolically linked named files.

```
edge cv
```

will archive symbolic linkage information only, while

```
edge chv
```

will archive the contents.

Note that this option does not affect symlinks to directories. **EDGE** will not traverse a symlinked directory because of the **h** option, but will instead back it up as a symlink normally. If you would like **EDGE** to traverse symlinked directories, use **-zFOLLOW\_SYMDIR** instead. Generally, you do not want either **h** or **-zFOLLOW\_SYMDIR** unless you know exactly what you are doing.

d Directory backup depth. This requires an argument specifying the directory depth level to be used when backing up. A level of 0 means only files in the current directory are backed up. No subdirectories are backed up. A level of 1 means only the first level of subdirectories are backed up. This option is automatically set to 0 when names are being piped in from another program using the **F** option...

```
find / -newer /tmp/newfile -print | \  
edge CvbkfF 64 149760 /dev/tape -
```

This prevents directories from being backed up multiple times.

R Unenforced READ LOCK. An attempt is made to READ LOCK every file. A READ LOCK means the file cannot be modified by any other program while the lock is in place. However, this lock is not enforced and if it cannot be obtained, the file is backed up anyway. This option is automatically assumed during **Master Backups** and **Differential Backups**.

Under Unix System V (which uses advisory locking), a warning message stating that the file was in use will be printed as follows...

```
WARNING: File was in use during backup
```

The file will then be archived. If the mandatory locking mechanisms are used on a file under BSD or SysV, and the lock cannot be obtained, the following warning message will be displayed...

```
WARNING: File was locked (System Enforced) will retry later...
```

and another attempt will be made to archive this file after all others have been archived. If the file is still locked with a mandatory lock on the second attempt, it is impossible to read and archive the file. Even the superuser cannot read it. Therefore the following message will be displayed...

```
edge: (second lock attempt failed) File was NOT backed up because  
lock was ENFORCED!
```

If the file cannot be locked, but can actually be read during the second attempt, the file will be archived and the following message will be displayed...

```
edge: (2nd lock attempt failed) File was backed up WITHOUT a lock!
```

If a lock cannot be obtained, the following warning message is displayed...

```
WARNING: File was locked (System Enforced) will retry later...
```

This file is skipped and then retried later after all other files are backed up. On retry, if the lock cannot be obtained (because it is locked by another program and has not been released), one of two events will occur; If the file can possibly be backed up it will, and the following message will be displayed...

```
edge: (2nd lock attempt failed) File was backed up WITHOUT a lock!
```

If the file cannot be read, or if **EDGE** would hang while trying to archive the file (because of a MANDATORY WRITE LOCK), the file will be skipped and the following message displayed...

```
edge: (second lock attempt failed) File was NOT backed up because lock was ENFORCED!
```

**RR** Enforced READ LOCK. This makes sure that every file is locked against modification before it is archived. The lock is guaranteed. If a file has already been locked against writing, **EDGE** will wait patiently for the file to be unlocked, then lock it against writing and archive it. A message that **EDGE** is waiting for the file to be unlocked is printed to the screen. When the lock is released, a message is printed that the backup is resuming.

This option supports Unix System V style locking as well as BSD locking. When a file lock is encountered, the following message is displayed...

```
^^^^--> LOCKED by another program!! Waiting...
```

When the lock is released...

```
Resuming backup...
```

will be displayed.

**RRR** Ignore All Locks. This option will override the default **R** (Unenforced Read Lock) option set in **Master Backups** and **Differential Backups**. It will cause **EDGE** to attempt to ignore all file and record locks and back up the file regardless of other access. This function only works on Unix systems where ignoring locks is permissible, and should not be used unless absolutely required. It is not possible to ignore locks in all cases, so some files might not be backed up.

- a Leave access time unchanged. This option will ensure that the file access time of any files archived will not be changed by **EDGE**. Normally the file access time is updated to the last time a file is read or archived. Unless this option is used, the access time a file will be changed each time it is archived. With the a option set, file access time will more truly reflect the times the files have been accessed by users. Some disk optimization systems make use of access times as part of their optimization algorithms. All files archived with this option will be backed up by

**Level 2 Differential Backups**, as their *ctime* access time will have been deliberately changed. If you are using the `a` flag, you should attempt **Level 1 Differential Backups** only.

**-zNODIR**

Normally, **EDGE** ensures that each directory entry itself, including the access permissions, group id and user id will be archived. This flag prevents directory entries from being archived.

**-zCOMPLEV=*n***

Enable software compression, using *n* (1..9) as the compression level. The higher the level, the smaller the backup will be, but the longer it will take. The default is 5.

**-zENCRYPT={filelist}**

If a license for the optional Encryption Module is present, then this will cause the files listed in {filelist} to be encrypted if they are archived. Note that these files must also be included on the command line or with a `-F` filelist, or be traversed over by something that is included.

**-zKEYBKP**

Normally, **EDGE** automatically excludes private decryption keys from backup of an unencrypted backup, or causes them to be encrypted if they are backed up as part of an encrypted backup. This option tells **EDGE** to treat private keys the same as other files.

**-zCHKSUM**

This enables data-level checksumming of the archive. By default, only headers are checksummed.

**-zBUFFERS={buffers}**

The default of 5 buffers allocated and used when the `S` modifier has been invoked may be increased to as large a number as your operating system allows, which may increase backup performance.

```
edge Mvbkfs 64 149760 /dev/tape -zBUFFERS=10 .
```

Unlike previous versions of **EDGE**, there is generally no need to tune the kernel to allow this option to work.

Additional buffers may also be allocated using the `ZBUFFERS` environment variable, which eliminates having to use the `-zBUFFERS=` flag on the command line.

```
ZBUFFERS=10
export ZBUFFERS
edge Mvbkfs 64 149760 /dev/tape .
```

**-zDEV={raw\_devices}**

This is the raw device backup option. A raw character device will be backed up from the start to the end. This is useful for large database applications that use raw partitions to store the data. The device specified must be a character device. If a block device is specified by mistake, it may result in an error message, unless the device's data can be archived. Any number of raw devices can be specified by using

a comma to separate one name from the other. There should be no spaces either before or after each comma. For example, the following command will perform a **Master Backup** and backup three large raw device data partitions.

```
Edge MVBkfs 64 149760 -zDEV=/dev/rhd10,/dev/ru2,/dev/ru3 .
```

The raw character devices will be backed up after all the data on the main filesystem has been backed up. An alternative way of specifying the command line is to enclose the raw device list in quotation marks separated by spaces:

```
Edge MVBkfs 64 149760 -zDEV="/dev/rhd10 /dev/ru2 /dev/ru3" .
```

If you perform a listing of files on the tape, the raw character devices will display as regular files with a large size. The size reflects the true size of the partition. This way the raw partition can also be restored by the regular **tar** utility.

You can set the environment variable **RAW\_SCRIPT** to the name of a program you want to run before each raw partition is backed up. This allows you to shutdown databases or other programs that may be accessing the partition. The script file is called twice for each raw partition, once just before the backup is done and once after the backup of the raw partition is complete. To distinguish the two, the first argument is either **-begin** or **-end**, so the script knows under which of the two cases it is called. The second argument is the name of the raw partition. The script should return with a zero exit code. If the return exit code is 100, **EDGE** won't backup the partition.

Using the above example, and assuming **RAW\_SCRIPT** is set to **/usr/lib/edge/bin/edge.rawscript** as follows:

```
RAW_SCRIPT=/usr/lib/edge/bin/edge.rawscript
export RAW_SCRIPT
```

the following actions/commands take place: (assuming **-zDEV="/dev/rhd10,/dev/ru,/dev/r3"** on the command line.)

```
/usr/lib/edge/bin/edge.rawscript -begin /dev/rhd10
```

Raw partition **/dev/rhd10** is backed up

```
/usr/lib/edge/bin/edge.rawscript -end /dev/rhd10
```

```
/usr/lib/edge/bin/edge.rawscript -begin /dev/ru2
```

Raw partition **/dev/ru2** is backed up

```
/usr/lib/edge/bin/edge.rawscript -end /dev/ru2
```

```
/usr/lib/edge/bin/edge.rawscript -begin /dev/ru3
```

Raw partition **/dev/ru3** is backed up

```
/usr/lib/edge/bin/edge.rawscript -end /dev/ru3
```

The example script **/usr/lib/edge/bin/edge.rawscript** is shipped with each copy of **EDGE**, and may be customized by the user.

#### **-zFOLLOW\_SYMDIR**

This option causes **EDGE** to follow symlinks to directories. Normally, **EDGE** backs up symlinks as a link rather than the underlying data. With the **h** option or -

**zFOLLOW\_SYMDIR**, it will back up the data in the regular file when it encounters a symlink to that file. With **-zFOLLOW\_SYMDIR**, **EDGE** will also traverse symlinked directories as though they are real directories. With **h** alone, **EDGE** will still treat symlinks to directories as links. This option implies **h**.

? This prints the version number of **EDGE** and a help screen.

---

## Modifiers used for Restore

- A** Absolute pathname strip option. This strips the leading slash “/” from the pathnames being listed, verified or restored, allowing files to be restored relative to the current directory. This also works for linked and symbolically linked files. When using this option, the user should always specify the pathname on the command line without a leading slash (/).
  - m** **EDGE** will restore modification time of file to current time rather than the actual file modification time.
  - p** Reserved.
  - h** Reserved.
  - w** Wait for confirmation. The user is prompted for a **y** on **n** answer on each and every file before. You must press **y** [Enter] for the file to be restored.
- Nw** This is the Non-destructive restore with interactive renaming. It is somewhat similar to the **w** modifier except it is only interactive when a disk file is about to be overwritten by a file on the tape. In this case, you choose to overwrite the file, skip over the file, or to rename the file. For example, if a file called `patient.dat` exists on your hard drive and you issue the following command:

```
Edge xvbkf 64 149760 /dev/tape -Nw
```

you see the following output:

```
patient.dat, 7 blocks
^^^^^^--> About to overwrite file!!
Size on hard disk:  3532 bytes  (Last modified: 07/26/97 at
02:20)
Size on tape :    3532 bytes  (Last modified: 07/26/97 at 02:14
Please choose:
  'o' - overwrite the file
* 's' - skip over the file and do not restore it
  'r' - rename the file and restore it under the new name
Enter choice:
```

In this way, you have full control over file collisions during a restore. If you choose **r**, you are prompted for a new name. You can use ‘~’ as the first character in a name to represent the previous directory name part of the file and thus save some keystrokes.

**-zWHERE={directory-name}**

Change the root directory of a restore. All files will be restored relative to the

directory given in the directory-name argument above. Subdirectories will be created as necessary. For example, if you had an archive containing the file `/usr/lib/edgefile` and wanted to restore it without overwriting the original file, you could do it like this...

```
edge xvbkf 64 149760 /dev/tape -zWHERE=/tmp ./usr/lib/edgefile
```

This would restore the file with the pathname `/tmp/usr/lib/edgefile`. This option internally invokes the **A** (strip absolute pathnames) option. Entire archives or directory trees may be restored in this fashion.

**-zDATEBEFORE={mm/dd/yyyy-hh:mm}**

Restores only files that were modified on or before the given date. This option requires an argument that specifies the EXACT date and time to be used. For example, to restore all files from an archive that were originally modified on or before June 14, 1998 at 4:00am, the following command line could be used...

```
edge xvbkf 64 149760 /dev/tape -zDATEBEFORE=6/14/1998-4:00
```

See "Specifying Dates And Times" more information on date and time specification.

**-zFLAT**

This is the flat file restore. The directory part of the specified files is stripped, and just the filenames are restored. The filenames are restored to the current directory. If the `-zWHERE={dirpath}` option is used, the files are restored to the specified directory path. You must specify the pathname (directory name and filename) of the files you want to restore. This pathname can either be absolute or relative depending on how the pathnames appear on the tape listing. If you just specify the filename part only, no files are restored. Otherwise, files overwrite on top of each other (e.g. README).

For example, if you want to restore files on the tape in `./usr/u/meddata` ending in `*.dat` to a new directory `/u2/meddata`, the following command is used:

```
Edge xvbkf 64 149760 /dev/tape -zFLAT -zWHERE=/u2/meddata \  
"./usr/u/meddata/*.dat"
```

The double quotation marks are necessary to prevent the shell from expanding the wild-card before it is evaluated by **EDGE**.

**-zTRIM14**

Setting this flag causes **EDGE** to truncate all pathname components, including subdirectories, to 14 characters. This allows archives made with newer systems to be restored on older systems which have 14 character filename limits.

**-zALL\_TAPE**

This causes **EDGE** to search the entire tape for files to restore, rather than quitting after it has restored the first copy of each file specified. This modifier allows proper restore of files with multiple instances on the archive.

**-zNOSEEK**

This causes **EDGE** to disable seeking during read, even if it thinks seeking is supported by the device.

#### **-zERROR\_RECOVERY**

This starts the restore in error-recovery mode, even if **EDGE** does not see a reason to do so.

#### **-zLEGACY**

**EDGE** will treat the archive as a legacy archive (from version 01.02.0x and earlier), even if it does not believe that this is the case. Normally, **EDGE** autodetects this.

#### **-zNOCHKSUM**

**EDGE** will ignore data-level checksums even if the archive contains them.

#### **-zREPLACE\_ALL**

This instructs **EDGE** to overwrite more files than it normally would during the restore. Normally, **EDGE** will not overwrite existing device nodes during a restore, nor will it convert one type of filesystem entity into another (for example, it will not replace a file with a named pipe). This option instructs it to do so if it is possible. Note that the nondestructive restore options, including restore-if-newer, are applied before this option takes effect; in non-destructive mode this option will have little or no effect on what is restored. Normally, you do *not* want to use this option.

#### **-zSEG\_NUM**

If specified, you can tell **EDGE** to start on a specific segment number rather than prompting for the segment to use interactively. For example, **-zSEG\_NUM=5** would cause edge to start on segment number 5. Generally, this is not necessary. **-zSLOTNAME** is generally a better choice under normal circumstances.

#### **-zMANUAL\_SEG**

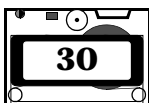
Normally, **EDGE** will be reasonably smart about which segments on a multi-segment archive will be displayed. It will also try to pick the next segment without user intervention if possible. For example, if an FTP backup is split into three segments, it will generally not display the second and third segments. It will also switch to them automatically as needed. With **-zMANUAL\_SEG**, **EDGE** will no longer do this, and will always prompt for which segment to use from all segments on the medium. This option is not necessary except in unusual situations.

## **4.5 Double Buffering**

---

The **s** (Speed) modifier initiates double buffering. This is the use of a complex scheme of shared memory buffers, and dual processes to significantly speed up the backup. While one **EDGE** process is busy compressing and packaging the data from the hard drive(s) into the shared memory buffers, the other **EDGE** process takes the data from the buffers and transfers it to the archive device. Because shared memory is used, there is no wasted time spent copying the data in memory.

The size of each shared memory buffer is the same as that specified in the blocking factor (**b** option). Therefore the optimum size of the shared memory buffer array can be



determined for your particular system by performing multiple backups, varying the blocking factor and noting the speed displayed in the backup summary.

The **s** option has been fully integrated into **EDGE** so that all features can be used, including compression to non-seeking devices (**P** option), multiple volumes, splitting files (regular or compressed) across multiple volumes and even the use of dual sequential archive devices.

By default, **EDGE** uses 5 memory buffers for double buffering. This can be changed using the **-zBUFFERS={buffers}** flag or the **ZBUFFERS={buffers}** environment variable.

In general, the speed improvements will be more impressive when compression is disabled and blocking factors of 128 or higher are used. The speed increases are most significant on computers with multiple CPUs and on those that do not already implement double buffering in the low level kernel tape device driver. In most cases, the double buffering feature will keep your tape drive streaming during the entire backup.

## 4.6 *Diagnostics*

---

The diagnostics are very thorough, with over 100 diagnostic, warning, or error messages. The user is notified if **EDGE** cannot obtain enough memory for read/write buffers as well as compression buffers. Command diagnostics are available to remind the user of the main options and what they do. **EDGE** notifies the user if the file size has changed while it is being archived and will automatically adjust for the new length while archiving. If the volume size is set and the end of the media occurs before the expected size given by the **k** factor, **EDGE** will notify with the message **UNEXPECTED: NO MORE SPACE**. Usually, **EDGE** can recover from this and will simply prompt for a new volume and continue backing up data. In any case, **EDGE** will suggest the correct volume size to use the next time. If a linked file is being restored and the file to which it is linked is not present on the hard disk(s), **EDGE** will notify the user of what file to restore first. This makes the task of restoring linked files easier and helps resolve link dependencies.

## 4.7 *Error Return Codes*

---

**EDGE** returns an error code indicative of what went wrong during the backup or restore process. A return code of 0 means the backup/verify/restore was successful. The following list shows the possible return codes and their respective meanings...

- 0 complete success
- 1 error in command usage or unlicensed product; a busy or unavailable archive device may also cause this error
- 2 miscellaneous error, not otherwise defined below
- 3 error reading from the archive device
- 4 error writing to the archive device
- 5 error opening or accessing a file or device

- 6 error while reading a file from the hard disk
- 7 error while writing a file to the hard disk
- 8 not enough memory available
- 9 read/write error in the Virtual Pipe (P option)
- 10 the header block of the file to be restored is bad
- 11 interrupted
- 12 error seeking on archive device
- 13 error while verifying data
- 14 byte level compare during level 2 verify
- 15 incomplete backup or restore; may also result from an expired license
- 16 problem with semaphores on system
- 17 problem with shared memory
- 18 dual process synchronization error
- [19 - 127] - (see the BackupEDGE User's Guide)
- 132 illegal instruction in the binary; most likely caused by using the wrong version of EDGE for a machine
- 136 floating point exception
- 137 program terminated with a kill -9
- 139 out of bounds memory fault; usually occurs on restore or verify of a compressed file when media is corrupt

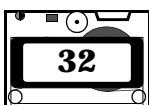
These errors must be distinguished from Unix system error codes. Unix error codes are noted in the catalog file, or on the screen in interactive sessions. The return code from **EDGE** will always be from the list above. However, the *EDGEMENU* menu system or the unattended backup system may issue additional errors, which are documented in the *BackupEDGE User's Guide*.

For instance, if you attempt to write more information than will fit on a single tape, Unix will typically return an error code 6 (No such device or address) indicating that it cannot write to the tape block requested. **EDGE** will report the Unix error code 6, then fail with an error return code of 4, meaning tape write error.

## **4.8 Compression Performance**

---

ASCII text files: Usually 40-70% compression. Database files: Usually 40-85% compression (best for larger files). Object/Compiled programs: Usually 20-60% compression.



Note that **EDGE** almost never causes a file to become much larger than it was originally, due to a very efficient compression algorithm. Even if a file is uncompressible by this algorithm, it will expand by at most 1%-2%.

The overall software compression ratio is provided in the summary. It measures the ratio of the output of the compression algorithm to its input. Files that are not compressed are not included in this ratio.

---

## **4.9 Backup Summary**

---

**EDGE** displays a summary of the backup process when the **v** or **V** option modifiers are invoked. This summary includes the date, the number of files archived, the total size of all files archived, the total archive time (not including volume switching), and the backup speed in bytes per second and megabytes per minute. If a volume size was used, the amount of room left on the current volume is also displayed. The number of files NOT archived is also displayed, as well as the number of files incompletely archived because of hard disk errors.

---

## **4.10 Restore Summary**

---

After a restore operation, **EDGE** displays the number of files successfully restored. In addition, the number of files not restored or incompletely restored is also reported. Files incompletely restored would include compressed files that could not be fully decompressed, virtual files that could not be fully re-virtualized, regular files in which a medium error was encountered and only part of the data was recovered, cross-device linked files that could only be made using symbolic links, and regular links used in place of unsupported symbolic links.

---

## **4.11 Level 2 Verify Summary**

---

After a **Level 2 Verify**, **EDGE** displays a summary including the number of files encountered, how many verified successfully, how many were excluded, how many did not verify, etc.

---

## **4.12 Volume Switching**

---

**EDGE** performs volume number checking on listing or restoring files. If the wrong volume is inserted, the user is notified and asked to insert the correct volume. **EDGE** allows the listing or restoration to start with any volume of the set. For the special case where one volume of a set has been lost, volume checking can be overridden by restarting the operation on the first out-of-sequence volume.

If one inserts bad media when switching volumes during a backup, **EDGE** allows the user to either try again, or to exit the program if it can recover. A simple menu guides these

choices. This is useful, for example, if one accidentally inserts a write protected tape after several volumes have already been backed up.

Sometimes the user may wish to quit the backup when a volume prompt is encountered. In these cases, the user can type a response of Q to quit.

During a backup to a CD/DVD resource, if no volume size has been specified **EDGE** will automatically detect it from whatever medium is loaded. Generally, it is recommended that you use identical media for each volume of a backup, but it is not strictly required.

## **4.13 Error Recovery**

---

### **Error Recovery During Backup Operations**

**EDGE** has several levels of error recovery. If the hard disk is failing and one must do a backup before replacing the drive, most backup programs will quit on the first bad file encountered. **EDGE** will switch into salvage mode and will ask if it should continue by displaying the following message...

```
edge: Hard error [error_number] in file [filename]
ERROR==> Files is incomplete!!
Do you wish to continue backup? (y/n)
```

If you answer with Y for yes, **EDGE** will replace the unobtainable data blocks with blocks of lower case x's. In these cases, it is advisable to use as small a block factor as possible, since the entire block factor of data is considered unobtainable and replaced with lower case x's. At the end of the backup the number of incompletely archived files will be reported. If these errors occur when **EDGE** is running as a background task, the yes answer will be assumed and as much data as possible will be archived.

### **Error Recovery During Restore Operations**

**EDGE** has two levels of error recovery on restoring data. The first is a hard error encountered while reading the media (bad sector or bad section of tape), and the second is a corrupt header block. Normally there should be a header block preceding each valid file. This block of data contains such information as the filename, file size permissions, and the date the file was last modified, along with link information. If either of these two situations occur, the user is prompted with the message...

```
Do you wish to attempt ERROR RECOVERY? (y/n):
```

A response of Y will cause **EDGE** to attempt to get beyond the bad spot on the device.

Error recovery works well for seeking devices. When dealing with non-seeking devices, one may have difficulty getting beyond the bad spot depending on the particular device driver and operating system. On seeking devices, **EDGE** will attempt to skip over the bad spot. A message indicating this seek attempt is shown. If this message is not seen and the device can seek then **EDGE** should be retried using the n option and a small block factor. The small block factor allows **EDGE** to skip in small jumps, thus capturing more valid

data. A diagnostic message indicating the possibly corrupt file will be shown. After error recovery, **EDGE** will automatically find the next valid file header and proceed extracting or listing files.

```
edge: directory not in proper format
      File header (name, size, date) is unobtainable
      This means that the header information about the file
      has been corrupted or is not in TAR format
Do you wish to attempt ERROR RECOVERY? (y/n):
```

If error recovery mode is selected (by pressing **Y**), the next valid file header will be found and the restore or listing will continue. This is useful for salvaging backups where one part of a multi-volume set has been lost or has become defective.

**EDGE** has a unique mechanism for finding valid files on an archive and is rarely fooled. It can also detect the true end of the archive and avoid restoring files on the media from older archives.

If corrupt data is encountered while decompressing a file, a message indicating that the data is corrupt will display. The remainder of the corrupt data will be skipped and **EDGE** will continue the restore starting with the next file on the archive. The corrupt file will be left on the hard drive and the data in the file will be intact up to the point where the corruption occurred.

## 4.14 *Dual Devices*

---

**EDGE** has dual device capability for handling dual tapes, dual floppy drives, or any combination of two drives. This option will archive to one drive then automatically switch to the other drive to continue the backup. This feature is useful for unattended backup of large amounts of data. If both volumes are complete, **EDGE** will prompt for further sets of dual volumes until all data has been archived. The dual device feature will also work with restoring files. This feature does not have a special command line option letter, but rather is invoked by using the **f** option twice. Each instance of the **f** option specifies one of the dual drives. The **k** option should be used twice to set the size of each respective volume. If the **k** option is used only once, **EDGE** assumes both volumes have the same capacity. For example,

```
edge Cvbkkff 64 249600 149760 /dev/tape /dev/tape2 /
```

will backup the entire file system using dual tape drives each with a different capacity tape. The dual device feature requires that the devices use the same blocking factor.

**EDGE** supports dual drive specification using the numbers 0-99 to represent the devices. For example,

```
edge MV -7 -8 .
```

will complete a **Master Backup** of all files using device entry 7 then automatically switching to device entry 8 when device 7 is full. The dual device feature does not strictly enforce volume number checking on restore.

## 4.15 Background Operation

---

**EDGE** is aware when it is operating as a background process. This is determined at startup and a message is displayed as follows...

### Running as a background task

There are several aspects that change when operating as a background process.

- **EDGE** will not go into Error Recovery mode. This is because Error Recovery mode is interactive with the user and it is not possible to do this effectively as a background process.
- If cancelled by the KILL command, **EDGE** will stop immediately. The usual popup menu that occurs when interrupted will not appear.
- If the user who invokes **EDGE** as a background process logs off, **EDGE** will continue to run.
- If **EDGE** encounters a bad spot on the hard drive while backing up a file or compressing a file, it will automatically answer all questions and obtain as much data as it can. The questions and automatic answers can be found in the catalog file.
- If **EDGE** needs to switch volumes, it will look for a reply in the pipe file `/dev/edge_listen`. **EDGE** will also look for answers to any question for which it does not have an automatic default answer (such as the format command when it encounters unformatted media) from `/dev/edge_listen`.

If a backup that is a background task takes more than one archive volume, after prompting for another volume, **EDGE** will wait for input from the pipe file `/dev/edge_listen`. The prompt message will appear as follows...

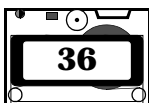
```
PLEASE REMOVE VOLUME : 1 INSERT VOLUME: 2 AND TYPE "y" WHEN
READY! __
(Put your reply in "/dev/edge_listen"   )
(   eg.  echo y >/dev/edge_listen     )
```

This message will appear in the catalog file if you have redirected process output and are using the `v` (cataloging) option. **EDGE** will wait patiently until a response is sent to the file `/dev/edge_listen`. As shown above the easiest way to send a response is simply to use the `echo` command. Place a new volume in the archive devices, then type the following from the shell..

```
echo "y" >/dev/edge_listen
```

**NOTE:** **EDGE** checks signal 2 to help it determine if it is running in the foreground or background. If you are trapping signal 2 within a shell or application program, **EDGE** will switch to background mode behavior.

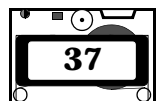
**NOTE:** **EDGE** also can be forced into background mode behavior by using the `-zBG` flag on the command line.



## **4.16 Signal Processing**

---

**EDGE** responds to the **QUIT** signal by immediate termination with a core dump. There is no special processing for this signal. **EDGE** responds to the **INTR** or **KILL** signals with the following popup menu:



Please choose:

- 1) Exit Immediately
- 2) Exit as soon as current file is finished
- 3) Resume execution
- 4) Exclude a file or new directory

Enter Selection:

The actions to be taken are as listed. Selecting option 4 shows:

NOTE - the current directory cannot be excluded

Enter name to exclude or 'q' to quit:

This allows a file or directory to be added to the exclude list which was overlooked when the backup was started. It is not possible to exclude the current file **EDGE** is working on, or the current directory **EDGE** is working within.

If the **KILL** signal is given to **EDGE** when running in background, it will immediately terminate.

## ***4.17 Screen Input And Output***

---

Under normal circumstances, **EDGE** will write to the standard output and receive input from the standard input.

If standard input is already being used to pipe filenames or data to **EDGE**, then the device `/dev/tty` is used for obtaining input from the user. If the standard output is being used for piping data to another program then screen output is sent to the standard error. Copyright information is always sent to the standard error.

If the program is being run as a background task from the Bourne shell or the Korn shell with job control turned off, the screen output is sent to the standard output. Screen input is obtained from the pipe file `/dev/edge_listen`. In these instances, prompts for input are followed with a short message notifying you to send your response to `/dev/edge_listen`.

If the program is being run from **cron** or from an **at** job, the output will be to the standard output and input will again be taken from `/dev/edge_listen`.

If the program is being run from the Korn shell with job control turned off, and is being run as a background task, output will be to the user's terminal and input will come from the user's terminal. The program will be suspended by the Korn shell when it tries to read from the terminal (and in some cases when it tries to write to the terminal). This is normal behavior for the Korn shell. The user will need to use the job control features of the Korn shell to move **EDGE** into the foreground and answer any questions. Optionally, the user can move the program into the background again at this point.

## 4.18 Use Of Wildcards

---

The wildcard characters \* (asterisk) and ? (question mark) can be used for pattern matching in most cases where a filename argument is required. The ? represents a single ASCII character. This can include an alphabetic character, digit, or punctuation mark. The \* represents any number of ASCII characters:

```
*.dat
*.idx
PAT?????.*
s.*
*cat.*
PAT??.DAT
PAT*.DAT
```

The wildcards can be used for pattern matching when specifying files to list, restore, exclude from a backup or restore, or exclude from compression. Wildcard expressions should be enclosed in either single or double quotation marks. This is so the shell won't expand the wildcard characters before they are passed to **EDGE**.

**NOTE:** The menu system traps shell expansion automatically. Do not use quotes when specifying wildcards in the **EDGEMENU** program, or in the file `/etc/edge.exclude`.

The following shows an example of the proper use of wildcards in a command that would restore all files ending in `.wk3` (spreadsheet files), except those files in directories that start with the numbers 1980 through 1989, which are excluded...

```
edge xvbkfE 64 149760 /dev/tape "198?/*" "*.wk3"
```

During a backup, it is sometimes desirable to omit the quotes, in order to let the shell expand the filenames automatically. This is not required, however, since any literal quotes will be expanded by **EDGE**.

There is some differences between shell expansion and **EDGE** expansion when dealing with certain patterns. In particular, patterns that do not specify a leading "/" or "." are treated by **EDGE** as "position-independent" patterns. This means that **EDGE** will search from the current working directory for all files that match. For example, the pattern `*.c` will match all files with names that end in `.c` that are in the directory from which **EDGE** is run, or any subdirectory. For example, the following command will archive `./config/devices.def`:

```
cd /usr/lib/edge
edge cvf /dev/null "*.def"
```

In contrast, without the quotes, the shell would expand `*.def` to match only those files in `/usr/lib/edge`. In this case, it is likely that no such file exists.

## 4.19 Excluding Files From Being Compressed

---

Files can be excluded from compression by any combination of the following three methods...

- 1 Using the executable bit.
- 2 Matching a filename suffix.
- 3 Pattern matching of file and/or directory names using wildcards.

All executable files (those with any execute permission bit set) are excluded from compression unless the **Z** modifier is used on the command line. Additionally, files can be excluded by filename suffix. By default, filenames with the following suffixes will not be compressed...

```
.Z .z .zoo .arc .zip .gz
```

Additional suffixes may be excluded from compression by setting the environment variable **SUFFIXES**. For example, to exclude files ending in **.gif** and **.lha** from being compressed, you could execute the following commands...

```
SUFFIXES=".gif .lha"  
export SUFFIXES
```

Suffixes must begin with a period (**.**).

Specific filenames and directories can be excluded by using the environment variable **COMP\_EXCL**. For example, if your filesystem contained graphic images contained in directories whose names always end in **.image** or **.graphic**, the following commands could be executed before running **EDGE**...

```
COMP_EXCL="*.image *.graphic"  
export COMP_EXCL
```

Up to twenty combinations of wildcard combinations or specific file or directory names may be specified in this environment variable. Each must be separated by a space or tab character.

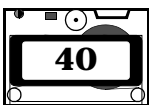
## 4.20 Specifying Dates And Times

---

Dates and times may be specified using one of two formats. Either format may be used with the **-zDATE** and **-zDATEBEFORE** option modifiers. The first format involves the traditional **MM/DD/YYYY** format, with optional time specification in 24 hour format. The complete specification is...

```
-zDATE=MM/DD/YYYY [ -hh:mm ]
```

where **MM** stands for month with 1 representing January and 12 representing December. **DD** stands for the date of the month. **YYYY** stands for the year. **hh** stands for hours (in 24 hour format) and **mm** stands for minutes past the hour. Hours and minutes are optional; if not used, time defaults to midnight of the specified date. Midnight is the beginning of the day, not the end of the day.



Preceding zeroes are optional. Here are some proper usages of the date format...

```
-zDATE=06/04/1998-18:33 (June 4, 1998 at 6:33pm)
-zDATE=6/4/1998-2:06 (June 4, 1998 at 2:06am)
-zDATE=2/4/1998-2:2 (February 4, 1998 at 2:02am)
-zDATE=1/31/2000 (January 31, 2000 at midnight)
```

The second valid format is the same as the date and touch commands on most systems...

```
-zDATE=MMDDhhmm[YYYY]
```

In this format, the year is optional. If not specified, the current year is assumed. All other date and time fields are mandatory and must be padded with zeroes if required. Here are examples of the same dates and times as above...

```
-zDATE=060418331998 (June 4, 1998 at 6:33pm)
-zDATE=06040206 (June 4, current year, at 2:06am)
-zDATE=02040202 (February 4, current year, at 2:02am)
-zDATE=013100002000 (January 31, 2000 at midnight)
```

February 4, 1998 at 2:02am could not be specified as `-zDATE=24221998`. Zero padding is mandatory when using this format.

Dates and times are checked for validity. For example, 2/30/1998 and 4/31/1998 would both be considered invalid dates. Hour must be a number from 0 to 24 and minutes must be from 0 to 60. When the date and time are converted to Unix format, time zones and leap years are taken into account automatically.

## **4.21 Virtual Files**

---

Virtual files are files whose reported size is much greater than the actual amount of data contained in the file. This is because the data not accounted for is null data, e.g. binary zeroes. Since the null data does not take up any real space on the filesystem, the file appears to Unix to contain more data than actually exists. The places in the file where the null data occurs will be referred to here as “black holes”. The ability of a file to contain “black holes” is unique to Unix.

If a virtual file is archived without any special attention, all of the null data will be read from the file and placed on the archive media. This is very inefficient and a waste of archive media capacity. Furthermore, when the file is restored, the null data will be restored as REAL nulls, causing the file to consume much more disk space than need be. After the restore, the file is no longer virtual; it is a very large file with null data in it. Therefore it is wise to mark virtual files for special consideration before backing them up, so that **EDGE** can both archive them efficiently and upon restore recreate the “black holes” so as to consume a minimal amount of disk space.

A list of all virtual files on a system should be placed, one per line, in a separate data file. The list may contain either absolute or relative pathnames; absolute is preferred.

The pathname of the data file should be placed in the environment variable `VIRTUAL_LIST`. This variable must be set and exported before running **EDGE**. It is not a bad idea to place this variable in your root `.profile` or `/etc/profile` file. For example...

```
VIRTUAL_LIST=/etc/edge.virtual
export VIRTUAL_LIST
```

Each file in this list will receive special attention during **EDGE** backups. During backup, **EDGE** identifies all of the “black holes” in the file while it archives the real data, plus “black hole” markers.

Upon restore, the data is decompressed if necessary, and the file is restored with all of the “black holes” placed exactly where they were originally. This process is known as “re-virtualizing”. It is not necessary to set the `VIRTUAL_LIST` environment variable during restores; **EDGE** knows when an archived file is virtual.

Virtual files may be identified by using the operating system’s `fsck` command. The command...

```
fsck [filesystem]
```

should be run on each UNMOUNTED filesystem and the output should be scanned for POSSIBLE FILE SIZE ERROR messages. For example,

```
fsck /dev/root
/dev/root
** Phase 1 - Check Blocks and Sizes
POSSIBLE FILE SIZE ERROR I=3042
POSSIBLE FILE SIZE ERROR I=8238
POSSIBLE FILE SIZE ERROR I=8596
POSSIBLE FILE SIZE ERROR I=9972
```

Would indicate that the files with the above four inode numbers are probably virtual. The `ncheck` command can be used to identify the filenames. In the above example,

```
ncheck -i 3042 8238 8576 9972 /dev/root
```

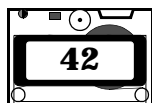
would find the actual pathname for each inode.

## **4.22 Network Backups**

---

The only requirements for network backup are a TCP/IP network and proper access permissions to the archive device on the host machine. It is not necessary to have an NFS network package.

**EDGE** allows backup of any node on a network to a host machine and archive device. **EDGE** will be executed using the CPU on the node to be backed up, and will establish a connection to the archive device on the host. **EDGE** will behave as usual except instead of sending data to the local archive device, the data will be sent through a network connection to the remote archive device. When software compression options are turned



on, the data on the local node will be sent across the network compressed which will reduce network traffic. Multiple nodes can be backed up to the same tape drive.

The following example will illustrate network backup. Let us say you want to perform a **Master Backup** backup of your work station and put the data on a tape drive on the host machine called `galaxy`. The name of the tape drive on the host machine `galaxy` is called `/dev/tape`. Suppose this tape drive has a 2 GB capacity. The command to do this from your work station is:

```
Edge MVbkf 64 1951488 galaxy:/dev/rmt0 .
```

**EDGE** will automatically make the network connection to the tape drive `/dev/tape` on the machine `galaxy` and perform the backup.

The following would do a backup to the resource `cdrom1` on the machine `mlitedfs`:

```
Edge MVf mlitedfs:cdrom1.
```

Note that the volume size and block size will be gotten from the resource.

If there were several nodes to backup, and the device `/dev/ntape` on `galaxy` was a no-rewind device, you could use `/dev/ntape` to backup multiple systems. In this way each node on the network could be backed up to a separate dataset of the tape. A simple shell script run from the host machine `galaxy` to back up the nodes called `alpha`, `beta` and `delta` would be:

```
for node in alpha beta delta
do
rcmd $node Edge MVbkf 64 1951488 galaxy:/dev/nrmt0 .
done
```

This would make each dataset on the tape contain the backup from each respective node. The first dataset would contain the data from machine `alpha`, the second dataset would contain the data from machine node `beta` and so forth.

Note that using the no-rewind node in this fashion is really not recommended.

The system name for the remote tape device may be specified by its node name, fully qualified host name, or IP address.

See the ENVIRONMENT variables `REMOTE_PUT` and `REMOTE_GET` for further information.

## 4.23 Environment Variables

---

The following environment variables can be used to help specify or modify the actions of **EDGE**...

### EDGEBIN

Specifies location of **EDGE** special binaries such as `edge.tape`.

### EDGECDAT

Specifies the catalog directory used when the `v` option is active.

#### EDGEFILE

Specifies the catalog file used when the **v** option is active.

#### SUFFIXES

Specifies suffixes of files to be excluded from compression.

#### COMP\_EXCL

Specifies files or directories to be excluded from compression.

#### COMPLEV

Specifies compression level, 1-9, and enables compression.

#### VIRTUAL\_LIST

Identifies the data file containing a list of files to be treated as virtual.

#### FIRST\_FILE

Identifies a file which will always be the first file archived on a backup set.

#### LAST\_FILE

Identifies a file which will always be the last file archived on a backup set.

#### INCREM\_TYPE

Sets the **Differential Backup** level to either **Level 1** or **Level 2**. If variable is not set, defaults to **Level 1**.

A **Level 1 Differential Backup** compares the file modification time (*mtime*) of each file against the start time of the last successful **Master Backup**. If *mtime* for the file is newer, then the file is archived. Modifications to a file which change *mtime* include creation, writing, and updating. Older releases of **EDGE** were only capable of **Level 1 Differential Backups**.

A **Level 2 Differential Backup** compares the file change time (*ctime*) of each file against the start time of the last successful **Master Backup**. If *ctime* for the file is newer, then the file is archived. Modifications to a file which change *ctime* include creation, writing, updating, moving, linking, and changing mode or ownership.

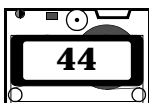
Some programs (like **tar** and **cpio**) purposely modify *mtime* when they restore a file. Therefore if a new program is installed, or if a file is restored from a backup, its *mtime* may be set to a time previous to the last **Master Backup**, which means that a **Level 1 Differential Backup** will ignore it. So a **Level 2 Differential Backup** is much more robust, although it may take up more archive space.

#### CHANGEDIR

Specifies a directory where a list of differences identified during **Level 2 Verification** is sent.

#### CHANGEFILE

Specifies a file where a list of differences identified during **Level 2 Verification** is sent.



The location of the **CHANGEFILE** can be set by using the variables **CHANGEDIR** and **CHANGEFILE**. Use **CHANGEDIR** to set the directory part of the path name. For example if **CHANGEDIR** is set as follows...

```
CHANGEDIR=/usr/lib/edge/lists
export CHANGEDIR
```

the detailed summary of files that change will be kept in the file

```
/usr/lib/edge/lists/ChangedFiles
```

If you then set the variable, **CHANGEFILE** as follows...

```
CHANGEFILE=doggy
export CHANGEFILE
```

the detailed summary of files that change will be kept in the file

```
/usr/lib/edge/lists/doggy
```

If **CHANGEFILE** is set but **CHANGEDIR** is not set, then the file will be kept in the **/tmp** directory. If the variable **CHANGEFILE** is set to the keyword **NONE**, then no record of the changed files will be kept. The output that would have been kept will go to **/dev/null**.

#### **BENOCHECK**

The variable **BENOCHECK** is used to specify a list of files that should not be checked. Please refer to "EXCLUDING FILES From Level 2 Verify" below for further details.

#### **SWITCH\_VOL\_CMD**

This allows you to customize what happens when you are to switch tape volumes. This specifies the actual program that is run when the end of the tape volume has been reached. The program that is specified is usually one that has been written on-site and is specific for your computer and tape drives. Instead of prompting you to switch volumes, the program you specify will be run. This is useful for large robotic backup devices that need special instructions. When this command returns it is assumed that the tape volume has been changed and that the backup or restore can resume. There will be no volume prompt message or intervention required when this command is used. It is assumed that the command specified will take care of whatever operator prompting is required.

As an example you could set the command as follows:

```
SWITCH_VOL_CMD="/usr/bin/tape unload /dev/xStp0"
export SWITCH_VOL_CMD
```

This would issue an unload command to the tape drive after each volume filled, which on many tape changers is sufficient to eject the current cartridge and insert the next available one.

The **SWITCH\_VOL\_CMD** can also send arguments to a custom designed script which include the device name and the volume just completed. **%v** is the volume number and **%t** is the device name. Consider the following commands:

```
SWITCH_VOL_CMD="/usr/lib/edge/bin/edge.special %v %t"
export SWITCH_VOL_CMD
cd /
edge Mvbkfs 64 149760 /dev/tape .
```

**EDGE** would start a Master Backup on `/dev/tape`. After the first volume was filled, **EDGE** would issue the following command:

```
/usr/lib/edge/bin/edge.special 1 /dev/tape
```

It would be up to the `edge.special` command to know what to do with the arguments `1` and `/dev/tape`.

When the command issued by `SWITCH_VOL_CMD` is completed, **EDGE** will continue the backup or verify.

If the command returns an exit status of `100`, **EDGE** will immediately stop.

#### REMOTE\_PUT

network command for writing to remote archive device or file.

This specifies the command to use to make contact with a remote tape device across the network. It can take two arguments that are represented by `%s`. These will be replaced by the hostname and the tape device to which you are sending data. For example, if the following command is used:

```
REMOTE_PUT="rcmd %s dd of=%s obs=64b ibs=64b 2>/dev/null"  
export REMOTE_PUT  
edge Cvf galaxy:/dev/rmt1 /etc
```

The actual pipe to connect to the tape device would be:

```
rcmd galaxy dd of=/dev/rmt1 obs=64b ibs=64b 2>/dev/null
```

This allows much flexibility in specifying network connections.

Users may, on some operating system platforms, use `rsh` (remote shell) or even `ssh` (remote secure shell) commands instead of `rcmd`.

#### REMOTE\_GET

network command to read from remote archive device or file.

This is similar to `REMOTE_PUT` above except that it is used as the actual command to obtain data from a network site for doing a restore or verify.

#### ZBUFFERS

The number of memory buffers allocated to the double buffering process. The default is 5.

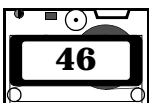
## 4.24 Excluding Files From Level 2 Verify

---

You can exclude files from a **LEVEL 2 Verification** by setting the variable `BENOCHECK` to the name of a file that contains a list of files to exclude. For example, if you set the variable as follows...

```
BENOCHECK=/etc/edge.nocheck  
export BENOCHECK
```

**EDGE** would look in the file `/etc/edge.nocheck` for a list of files to exclude from byte by byte comparison. This variable is only active during a **LEVEL 2 Verify**. If the file `/etc/edge.nocheck` contained a single name `./etc/wtmp`, then this file would be



excluded from byte level comparisons. Since this file changes every time someone logs into the system, it will often be different from the file on the tape. Most systems have a standard set of files that change on a minute to minute basis, and these are the files to put in this list.

The list should be structured so that there is a single file on each line. If your backup was done with a leading `./` then the name of each file in the list should likewise start with a leading `./`. If the list includes a name of a directory, the entire directory and all subdirectories will be excluded from the verification. When a file is excluded, you will see the message...

```
./etc/wtmp, 5 blocks <Excluded>
```

as **EDGE** is reading the archive media. This file will simply be excluded from checking during **Level 2 Verification**. It will still be checked at **Level 1 Verification**. Thus you can limit your **CHANGEFILE** file to a list of files that are potentially serious. Additionally, when **Level 2 Verification** is done at night, the error code is more likely to be a true reflection of files that really did not verify, as opposed to those that were modified by the operating system.

## 4.25 Files

---

**/dev/edge:**

used as the default device if neither an **f** option and argument or a device number **0-99** is specified. This should be linked to the actual device to be used. **EDGE** assumes this device can seek.

**/dev/edge\_listen:**

**EDGE** uses this pipe file to communicate with background tasks.

**./Master\_backup:**

**./etc/Master\_backup:**

These files hold the date of the last **Master Backup**.

**/etc/default/edge:**

The file containing the library of available devices.

**/etc/default/tar:**

The library file used if the above file is not available.

**/usr/lib/edge/lists/:**

Default directory to store catalog files.

```
$EDGE/CAT/Backup_{month}.{day}:
```

```
$EDGE/CAT/Master_{month}.{day}:
```

```
$EDGE/CAT/Increm_{month}.{day}:
```

```
$EDGE/CAT/Verify_{month}.{day}:
```

```
$EDGE/CAT/Listing_{month}.{day}:
```

```
$EDGE/CAT/Restore_{month}.{day}:
```

These are the names of the catalog files for each respective function. **EDGE**CAT is a shell variable which can be set to the directory that should contain these files.

**\$EDGE**CAT/LAST\_Backup:

**\$EDGE**CAT/LAST\_Master:

**\$EDGE**CAT/LAST\_Increm:

**\$EDGE**CAT/LAST\_Verify:

**\$EDGE**CAT/LAST\_Restore:

**\$EDGE**CAT/LAST\_Listing:

These files hold the catalog of the last set of files that were archived, restored, or listed. Each file is physically linked to a corresponding catalog file whose name ends with a month and day.

**EDGE** keeps **Level 2 Verify** detailed information in the **CHANGEFILE** called /tmp/ChangedFiles. Alternatively if the environment variables **CHANGEFILE** and **CHANGEDIR** are set the detailed information will be kept in **\${CHANGEDIR}/\${CHANGEFILE}**.

## 4.26 *Limitations*

---

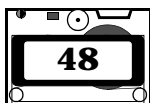
The total length of the path and file name is limited to approximately 5,000 characters. If a path exceeds this a warning message is printed, and the backup resumes. If a file is shortened while being archived, **EDGE** will fill the difference between the new and old size with nulls. To extract special character/block device files or to create a new directory that did not previously exist, **EDGE** must be invoked by the super user. **EDGE** does not support the **u** or **w** options of **tar**.

**EDGE** from the command line can be used to write directly to a CD-R, CD-RW, or DVD device, or anything that **EDGEMENU** can write to. Specify the resource name with the **f** option. Omit the **k** and **b** options when doing this unless you want to override the resource settings.

If **EDGE** is used on volumes out of order during a **Level 2 Verify**, it will start verifying on the first file that starts on the volume.

The maximum capacity of a seeking device is limited to the operating system **ulimit** for a single file. Note that if you are writing to an FSP or URL resource, **EDGE** will automatically split the archive up into multiple segments as required, as specified by the resource's segment size. By default, the size per segment is slightly less 1 GB. Up to 255 segments may be used.

**EDGE** does not update or use data found in Backup Sequences, nor does it support **Incremental Backups** without using the **-zDATE** flag. Only **Master Backups** and **Differential Backups** are directly supported.



## 4.27 *Compatibility*

---

**EDGE** can read, verify, and bit level verify archive media created by most versions of **tar**. **EDGE**'s unique restore and error recovery capabilities can be used on any single or multi-volume **tar** archive.

**Tar** can generally read floppies / tapes produced by **EDGE**, although files that span volumes might not be recoverable easily. However, files compressed by **EDGE** will remain compressed. A file with a pathname greater than 100 characters in length cannot be properly restored by most **tar** programs. Only **EDGE** can properly restore this file. If you attempt to restore such a file, its filename will be hard to predict, although **EDGE** takes care to make sure that it is well-defined. Generally, it is a good idea to treat such files as unrestorable except by **EDGE**.

Using the standard UNIX program **pax** to restore **EDGE** archives will do slightly better than **tar** in some cases, especially with longer filenames. Compressed, virtual, and/or encrypted files cannot be restored by this utility easily, however, since **pax** does not support those features.

Versions of **EDGE** prior to 02.00.00 have similar restrictions as **tar** when reading **EDGE** archives made by **EDGE** version 02.00.00 and later.

**EDGE** version 02.00.00 and later can read older **EDGE** archives transparently.

## 4.28 *Special Internal Features*

---

**EDGE** can handle an almost unlimited number of links and symbolic links during backup. This is due to a very space efficient method of storing these in memory.

The **ulimit** (maximum file size) variable is checked and automatically increased if necessary.

During data compression and decompression, there are no intermediate files. A compressed file is expanded in a streaming fashion as it is read from the archive media to the hard disk. This ensures that there is no disk fragmentation during the restore process. Data de-compression also occurs in a streaming fashion, with no intermediate files. The data compression and decompression functions are built directly into **EDGE** with no reliance on external programs.

## 4.29 *Using EDGE With The Menu Environment*

---

The normal **EDGE** binary is **/bin/edge**. In addition, a shell script called **/bin/Edge** is provided. This shell program sets and exports all variables set in **/etc/default/edge.cfg** before calling **/bin/edge**. Consider the following two commands:

```
/bin/edge MPZvbkf 64 149760 /dev/st0 .  
/bin/Edge MPZvbkf 64 149760 /dev/st0 .
```

The first would use all of the defaults for directory names, files to exclude from compression, etc. that are built into the **EDGE** binary, while the second would use the values set in the `edge.cfg` file. f

### ***4.30 Standards Conformance***

---

**EDGE** is not part of any standard. It is provided by Microlite Corporation only for use with **Microlite BackupEDGE**.

### ***4.31 Disclaimer***

---

Microlite Corporation provides this software AS IS without any warranty of any kind, and is not responsible for any direct, indirect, or consequential damages that may arise due to use of this program.

## 5 Hints And Useful Tips for /bin/edge

---

### 5.1 Backing Up And Restoring

---

The following discussion is not directed at users of **EDGEMENU** or **Scheduled Jobs**. It is intended only for those using **/bin/edge** directly.

**EDGE** makes the ordinary task of archiving data an interesting and enjoyable process. Several items of general interest regarding command line options should be discussed to understand the full power of **EDGE** and to avoid potential pitfalls. First, **EDGE** uses the standard Unix option argument format. This format is...

```
commandname -optionlist option-arguments
```

It is essential that the option characters be given the same order as their respective arguments. If this is not the case, then the arguments will be misinterpreted. The '-' dash symbol preceding the option list is optional.

For example, to perform a **Master Backup** to a tape device **/dev/tape** with a block factor of 64 and capacity of 150 Megabytes the following command is used...

```
edge Mvbkf 64 149760 /dev/tape .
```

Please note that the arguments follow the options in order. If you were to reverse the order of the options you would need to reverse the order of the arguments. The following commands are all equivalent to the one above...

```
edge Mvfbk /dev/tape 64 149760 .
edge Mvbkf 64 /dev/tape 149760 .
edge Mvbkf 149760 64 /dev/tape .
```

Because the number of options and arguments can get large, there is an alternative method of specifying options. Using the above example, the alternative method is...

```
edge Mv -f /dev/tape -b 64 -k 149760 .
```

Keep in mind, that you must use a leading '-' before each additional option letter with its respective argument. This is so that **EDGE** can differentiate between options, arguments, and filenames.

It is important to remember to use the same options in listing, verifying, or extracting that were used in backing up. This means the volume size and block size options **k**, and **b**, as well as the seeking option **n** and the file device option **f**. These options specify the edge format to be used and if not used consistently with the same volume set, erroneous results will occur when crossing volumes. It is good practice to record the exact options used directly on the archive media label. If omitted from the command line during a read operation, edge will generally use the volume size from the label automatically.

Restoring files from **EDGE** is very simple. The **Master Backup** is restored first. This brings the filesystem up to the date of the last **Master Backup**. Then the last **Differential Backup** is restored. This overlays all files that have changed with the

current version of these files. All files, empty directories (those with no files), device files, pipes, links, and symbolic links are restored to their previous status. This obviates the need for remaking links, remaking device files, or redoing the port configuration. It also makes disk swapping and filesystem swapping between similar machines much easier.

Keep in mind that if you are ever doing a non-destructive restore (using the **N** option), you should reverse the order of restoring the **Master** and **Differential** backups. You should first restore the **Differential Backup** and then following this, the **Master Backup**.

For most systems, doing the **Master Backup** and **Differential Backup** from the root directory will suffice. If the system is large (containing many mounted filesystems), it may be preferable to do a **M** or **I** backup on a particular filesystem. There must be a `./etc` directory on the local filesystem so that `./etc/Master_backup` can be made. If there is not, **EDGE** will make the file `./Master_backup` in the current directory to hold the backup date. Then with the filesystem mounted (e.g. on `/usr2`) one would use the following shell script...

```
cd /usr2
edge Mvbkf 64 149760 /dev/tape .
```

Then any subsequent **Differential Backup** from this mount point (`/usr2`) will only backup files changed since the date of the last **Master Backup** as specified in the file `./etc/Master_backup` or `./Master_backup`.

If this technique is chosen, one should exclude this filesystem when doing the **Master Backup** from the root directory. Otherwise the filesystem mounted on `/usr2` (in this example) will be backed up twice. Using the example above, the command to do this would be...

```
edge MvbkfE 64 149760 /dev/tape ./usr2 .
```

One advantage of this technique is that other users could be using the files in `/usr2` while the backup is being made.

For brevity, the **b**, **k**, and **f** options and their arguments will be eliminated from many of the following commands, so that it is easier to quickly see the options and arguments under discussion.

## 5.2 *Backing Up Select Files*

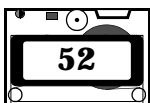
---

Sometimes one would like to archive only select files in a list or a directory. The following simple command shows how this can be done using **EDGE**. The list can be produced by an editor or with a command such as...

```
ls . > list
```

Then the file list can be edited or modified. Once the list is satisfactory, simply run the following command...

```
edge cvF list
```



The **F** option tells **EDGE** to look in the file specified by the argument for the list of names. The list should be one name per line and can be any length. To add files to the end of a floppy diskette archive without compressing, one would use...

```
edge rCvnbkfF 36 1440 /dev/rfd0135ds18 list
```

---

### 5.3 *Data Compression*

---

**EDGE** can compress files during archiving. If given a resource name as the destination, it will default to the resource's setting for compression. If given a device node, then it will default to not compressing data unless at least one of the **n** (seeking) and **P** (pack-regardless) options are also present.

**EDGE** compresses files without making a temporary file. The file is actually compressed as it is streamed through **EDGE** and the compressed output goes directly to the backup media. This allows for exceptionally fast compressed backups.

For example, the following two commands will perform compression...

```
edge cvnbkf 36 1440 /dev/rfd0135ds18 files...
edge cvPbkf 36 1440 /dev/rfd0135ds18 files...
```

To absolutely turn off compression use the **C** option. Example...

```
edge Cvf floppy0 files...
```

To compress only very large files use the **L** option (limit) as follows...

```
edge cvLf 100 floppy0 files...
```

Thus, **EDGE** will only go into compression mode for files greater than 100 (512 byte) blocks. All smaller files will be backed up without compression.

An astute observer will notice that **EDGE** will spend more time compressing "dense" files than "loosely packed" or redundant files. This is because **EDGE** is trying to find a way to make the "dense" files even more dense and this takes a little more time. Do not be alarmed.

Exactly how hard **EDGE** works to compress files can be modified with the **-zCOMPLEV** option:

```
edge cvPf /dev/null -zCOMPLEV=9 files...
```

This will compress files as much as **EDGE** can, although it will take significantly longer to do so than the default setting of 5. This compression level can also be set via the **COMPLEV** environment variable.

---

### 5.4 *Miscellaneous Tips*

---

---

#### **Backing Up From the Root Directory**

A common mistake when backing up from the root directory is to use...

```
edge Cvf /dev/some_device *
```

instead of

```
edge Cvf /dev/some_device .
```

The asterisk will match all files in the root directory except those starting with a leading period. Thus, your valuable `.login`, `.exrc`, `.cshrc`, and `.profile` files (among others) will not be backed up. If your system crashes and you need to restore, you will not get a complete restore and you will need to recreate these files.

Another common mistake is to use the slash “/” as in...

```
edge Cvf /dev/some_device /
```

This will backup all the files using a leading slash “/”. This is referred to as an absolute pathname format. This makes it somewhat more difficult to restore an individual file in a different directory than the one from which it came. You are forced to use the **A** option to do this. For example, you might wish to restore one of the files into the `/tmp` directory to examine it. If you forget to use the **A** option during the restore, then the file will be restored to its original location on the hard drive.

For this reason, it is recommended that backups from the root directory start with a period “.”. This puts the pathnames in relative format. It is much easier to restore the file to a different directory when files have been backed using the relative pathname format. For example, if the file you want to examine is `/usr/bin/chapter1`, you would do the following to extract it without wiping out your current version.

```
cd /tmp
edge xv ./usr/bin/chapter1
```

In this way the file will be available in `/tmp/usr/bin/chapter1`. Using a leading period “.” when backing up always leaves you with more options when you begin to restore.

---

## Saving Screen Output in a File

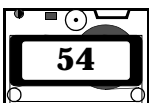
You may encounter a situation in which you want to have the screen output of **EDGE** go simultaneously to the screen and also be saved in another file for later inspection. This is easy when the **v** option modifier is used as it creates a catalog file. The catalog file is a carbon copy of the screen output. You can customize the location and name of the catalog file by using the environment variables **EDGECAT** and **EDGEFILE**.

**EDGE** sends all warning messages, error messages, and the filename list to the standard output.

---

## Converting Catalog Output to a File List

If you want to take a catalog file and convert it to a raw list of files, you can do this using the `edge.cnvt` utility. This will take any type of catalog file (backup, verify, list, or restore) and convert it to a raw list of files. This list can then be suitable for editing or to create a list for input to backup or restore with the **F** option. This utility



resides in the `/usr/lib/edge/bin` directory. For example, to convert the file called `LAST_Master` to a raw list called `/tmp/filelist`, the following command could be used...

```
/usr/lib/edge/bin/edge.cnvt LAST_Master > /tmp/filelist
```

---

## Extra Volume Prompt

In rare instances you may notice **EDGE** prompt for another volume but not write any information on it. This may occur when the files you are backing up take up exactly the storage capacity of the archive device. The same will occur when listing or extracting the files. Do not be alarmed if this occurs.

---

## Negatively Compressed Files

Another problem occasionally encountered is negative compression of a file. While the compression algorithm is quite efficient, and tends to expand files by no more than 1-2%, if you have a very large number of uncompressible files then it can still be a good idea to exclude them from compression for a net space and time savings. Files can be excluded from compression in any of the following three ways...

- 1) Matching a filename suffix.
- 2) Pattern matching of file and/or directory names using wildcards.
- 3) Using the executable bit.

First, if the file ends in a unique suffix, you can exclude all files ending in this suffix using the `SUFFIXES` environment variable. Second, you can exclude the file specifically by putting it in the environment variable `COMP_EXCL`. Alternatively, you can exclude the parent directory of the file by including it in `COMP_EXCL`. This will exclude all files in the parent directory.

Third, you can change the permission mode of the file to indicate it is executable.

```
chmod 755 filename
```

Unless overridden with the `z` option, **EDGE** will not compress any file that has an executable flag in its permissions.

Note that changing the permissions on a file simply to have it excluded from backup is generally not the right idea. It is presented here for completeness only.

Please refer to “Excluding Files From Being Compressed” on page 37 for more details.

---

## 5.5 Exchanging Tapes With Other Computer Systems

The most common problems encountered arise because the same block factor and capacity was not specified on both ends. If these two items are taken into careful consideration you should not experience any problems with data transfer between systems.

### 4mm or 8mm Tapes

A big problem in transferring data on these tapes is that there are two modes in which these tape drives can write data. Each method is incompatible with the other. The tape drive is jumpered for one of the two methods when shipped. The first method is called fixed-block-mode and the second is called variable-block-mode. Although the tape drive is jumpered for one mode or the other, the modes can also be changed through software. SCO Unix (prior to OpenServer 5) SCSI tape drivers use the fixed-block-mode as the default and SUN OS tape drivers use the variable-block-mode as the default.

If you send an 8mm tape written by a tape drive in variable-block-mode to a computer whose tape drive uses fixed-block-mode you simply will not be able to read the tape. The modes are incompatible with each other.

If you try to read a tape written in fixed-block-mode on a tape drive that uses variable-block-mode, it can only be done using a block factor of 2.

If you try to read a tape written in variable-block-mode with a tape drive that uses variable-block-mode, the only way to do this is to use the same exact block factor on the restore as was used on the backup. Otherwise, you will receive SCSI error messages about illegal length.

There is only one way a variable-block-mode tape drive can transfer data to a fixed-block-mode tape drive. If the backup on the variable-block-mode tape is done with a block factor of 2, it can be read with a fixed-block-mode tape drive using the same block factor. This is the one exception to the rule stated above about variable-block-mode and fixed-block-mode being totally incompatible.

## **5.6    *Trouble Shooting Common Problems***

---

### **Unattended Backup Fails**

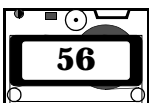
There can be several reasons for the failure of an unattended backup. The most common problem is that the `crontab` utility stops running. Although in theory, this should never happen, in practice it does. You can make sure that the `crontab` utility is running by using the following command...

```
ps -e | grep cron
```

If you see a line containing the word `crontab` (excluding any line for the `grep` command itself), then it is running. If you see nothing, then the `crontab` utility is not running. You should check the clock specification file to be sure it contains the command line invoking the unattended backup. The command `crontab -l` (that's a lowercase `l`) will display it.

You should check your system clock using the `date` command. The system clock could be set for the wrong time. This is especially possible after a hardware crash or change of a system component.

You should check the mail as the `root` user. If the backup starts and fails, there will usually be mail sent to the root user. Alternatively, there may be a message on your default system printer.



You can see if **EDGE** is already running. Sometimes, the unattended backup will fail because **EDGE** is already running and hung up for one reason or another. Also, the unattended backup from the night before may have filled one tape volume and be waiting for a reply after you switch tape volumes. The command to see if **EDGE** is already running is...

```
ps -e | grep edge
```

You should check to sure that you can invoke **EDGE** manually and that it runs okay. Be sure to use the same parameters the unattended backup would use.

### **Tape Light Stays On, But No Tape Activity**

This is referred to as a hung tape drive. This results when the tape drive has finished writing a chunk of data to the drive and signals the tape driver that it is ready for more data by issuing an interrupt. But the interrupt is never received by the tape driver. This can happen with an interrupt conflict. The most common interrupt conflicts are with serial port (interrupt 3) or with a second parallel port (interrupt 5). In these cases, the driver for the serial port or second parallel port receives the interrupt instead of the driver for the tape drive.

Under SCO you can check for possible conflicts by running the command `hwconf ig`. This will show you the settings of your currently configured hardware. Look at the interrupt settings for any conflicts.

When you have a hung tape drive you cannot use the [Del] key to interrupt **EDGE**. In fact, you cannot even kill **EDGE** with a `kill -9` command. The reason for this is that **EDGE** is in kernel mode waiting for the tape driver to return and the tape driver is waiting for the interrupt from the tape drive. Sometimes, taking the tape out and putting it back in will make the tape drive usable again. Sometimes turning the unit off and then on again will be useful. Sometimes issuing another command that accesses the tape device will be useful. There are situations, unfortunately, in which the only solution is to re-boot the computer.

### **Cannot Kill EDGE even with kill -9**

This happens when the tape drive is hung as explained above. **EDGE** has been put to sleep by the kernel while it is waiting for the tape driver to return. Since the kernel tape driver routine is hung waiting for an interrupt that will never occur there is nothing that can be done short of making the tape drive generate an interrupt. This can be done by taking the tape out, turning it off and on, putting another tape in, or trying to access the tape device from another terminal.

This situation is not **EDGE**'s fault. It is in kernel context with a hung tape driver and cannot be killed. The same problem can occur with `tar`, `cpio` or any other utility that uses the tape driver.

### **Tape Verified Fine, but when Restored, an I/O error 5 occurred**

In this case, the tape was probably exposed to electromagnetic radiation between the time it was verified and the time it was restored. The most common cause for this is the tapes are stored near the computer monitor or on a shelf just above a

computer monitor or terminal. The electrostatic surge of turning the monitor or terminal on or off weakens the magnetic charge on the tape and introduces an error. All tapes and other magnetic media should be stored at least 16 inches away from any electromagnetic fields. For stronger electromagnetic fields, this distance should be increased.

## Distinguishing Software Errors from Hardware Errors

There are two types of basic errors that can occur when restoring data. The first is a software error. This means that there is data returning from the archive device, but the data is not in the format expected by the backup program. This can happen if the data on the archive media was overwritten or partially overwritten by another piece of software. Also, the data may have been written in a different format.

One classic example of this occurs under the SCO XENIX or Unix operating systems. Data is backed up using `/dev/erct0` as the backup device. This device does error correction and pads the data on the tape with additional error correction information. Then someone decides to restore the data using `/dev/rct0` as the tape device. Because this device does not understand how to interpret the additional error correcting information, this information is returned along with the actual data. The combination of two types of data causes **EDGE** to realize the data is not in proper format. It then aborts with an error message as follows...

```
edge: directory not in proper format
File header (name, size, date) is unobtainable
This means the header information about the file
has been corrupted or is not in the TAR format
```

If you see the above message, you know you are dealing with a software error and that the data on the backup media has somehow become corrupted or was never in the `tar` format. The integrity of the media, however, is okay. You do not need to get new floppies or tapes or replace the tape drive.

A hardware error, on the other hand, presents itself in a totally different fashion. You will receive a message about an I/O error. In most cases this will be an error 5. This means that the media has been damaged, either physically or electromagnetically. Alternatively, your tape drive or floppy drive or cabling may have a hardware problem.

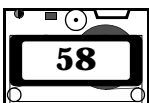
```
edge: Tape Write ERROR 5 occurred because I/O error
Offset is 2000K
Room left is 42000 K
```

This means that the data simply could not be read from the device. This is a hardware error.

## Directory not in Proper Format

This really means that the data on the archive media is not in the `tar` format. The word “directory” is confusing. In this context, it refers to the directory of information on the media referring to files. This information has been scrambled, corrupted or was never correct.

## Ran out of Space during a Restore



## Same filesystem.

If you run out of space restoring data to the same filesystem from which it was backed up, there can be several explanations. First, a large file may have been created on the system between the time you backed up and restored. Most commonly, this will occur when you clean the system using `fsck` and orphan files are put in the `./lost+found` directory. Check this directory and remove any files that are not needed. Second, you may have intended to backup only one filesystem but in fact backed up much more than this. Check the listing of the tape and see if there are files on it that you did not intend to back up. Third, if the filesystem you are restoring to has any filesystems mounted on it, the mount may have failed and you don't really have any filesystems mounted. You can check this with the `mount` command to be sure the proper filesystems are mounted.

Fourth, you may have a problem with virtual files that were not marked and backed up as special virtual files. In this case, they will occupy much more space on the hard drive after the restore than they ever did before the backup. The clue to this problem is the size of the file. If it is very large it could be a virtual file. Your best bet is to exclude the offending file from the restore. Then, take the tape to another system with more space and try to restore the virtual file. You should review the section entitled **VIRTUAL FILES** in the manual to see how to properly backup virtual files so that this will not happen again.

## Backed up a file, but EDGE cannot find it when I Restore

There are several reasons for this type of problem. Most commonly you have misspelled the name when you attempted to restore the file. If you are using the `F` option and have put the filename to restore into another file, you may have accidentally put a space character or two at the end. Try deleting the name and re-enter it being careful to end the filename properly.

You should list the files on the media to be sure the one you want is there. Be sure you are using the Unix forward slash `'/'` in the pathname rather than a DOS backslash `'\'`. When working in both environments it is easy to get confused.

When all else fails and you see the file listed but cannot get it off, the following technique has proved to be very useful...

```
EDGEFILE=/tmp/List
export EDGEFILE
edge tVF /dev/tape > /tmp/List
/usr/lib/edge/bin/edge.cnvt /tmp/List >/tmp/List2
vi /tmp/List2 (or whatever word processor you like)
```

remove all files except the one you want

```
edge xvF /tmp/List2
```

The filenames are captured into a catalog file and this file is converted to a raw list of names. At this point the spelling is guaranteed to be correct. All other files from the listing are then removed. Finally, this list which contains the files of interest is used as the list from which to restore. This always works!

## There are Stp (tape) Error Messages on the Console in the Morning

Some drivers report error messages about the SCSI tape not being ready and these appear on the console in the morning after an unattended backup. This results from the SCSI requirement to reset a SCSI device twice after certain commands. Some tape drivers don't realize the SCSI device needs to be reset twice and report the initial failure as an error message. Ultimately, a repeat reset is done and all is well.

### **After a Full Restore the Master Backup Date is Incorrect**

This is a side effect of the way the **Master Backup** date is updated. Since it is updated at the completion of the backup (only if successful), when the file `./etc/Master_backup` is backed up, it still has the old date in it. When you do a full restore, the old date is restored.

The solution is to use the `touch` utility to adjust the date as follows...

```
touch /etc/Master_backup
```

This solution should be performed immediately after the full restore before any files have been modified. Alternatively, after the full restore and any adjustments have been made, you can run a fresh **Master Backup**.

### **Restore Prompts for more Tape Volumes than Exist**

This will happen if you use a smaller capacity for the tape than was used to back it up originally. The most common example of this is a 150 megabyte tape being restored as a 60 megabyte tape. After only 60 megabytes of the 150 megabyte tape have been read, you will be prompted to switch tape volumes. If you are not sure what capacity was used when the tape was backed up, it is best not to use any capacity when you restore. **EDGE** will determine when it needs to switch volumes. If a file is split across tape volumes, **EDGE** will switch after it reads the first part of the split file. Otherwise, **EDGE** will stop at the proper end of the tape and will require another invocation of the command line to read each subsequent tape.

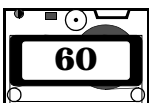
### **Restore Prompts for another Tape Volume before it should**

This is similar to the above problem. A smaller tape capacity was specified on the restore than on the original backup. When you insert a second tape, you will usually receive a "Directory not in proper format error".

### **After a Full Restore SCO 'fixperm' Utility Finds Problems**

This is similar to the above problem. The SCO `fixperm` utility will compare a static database of file permissions and ownerships to the actual files on the hard drive. Any differences are noted and modifications are made if desired. However, if your backup was done in multi-user mode rather than single user mode, you will note differences caused by daemons and other software that is running in multi-user mode. For example, the print spooler package will change the permissions of certain spooler files while it is running and change them back when it shuts down.

For this reason, the SCO `fixperm` utility may find some differences. These are usually insignificant. They do not reflect a problem with the way **EDGE** restores file permissions. If you believe you have a problem, you should run `fixperm` in single-user mode before doing a **Master Backup** in single-user mode. Then do a full



restore to a fresh hard disk. Then run `fixperm` again and you should see no differences. You will note that all permissions and ownerships will be the same.

### **After a Full Restore some File Permissions Appear to have Changed**

**EDGE** will reliably restore the file permissions to what they were when the system was backed up. There are some files that change owner and permissions on a regular basis when the system is in multi-user mode. The `login` program, for example, will change the owner of the terminal to that of the person logging in. Also the permissions of the terminal are changed. UUCP will change the permissions and ownership of dial out ports while they are in use.

Therefore, any differences that are noted are those that normally occur on an active multi-user system. If you run your backups while in single-user mode, the restored permissions and ownerships will be identical to the single-user mode filesystem.

### **After a Full Restore the Catalog File is Missing Files**

This can be noted after a full restore. If you examine the catalog file that reflects the backup that was just restored, only about one half the files are present. This is because at the time the catalog file itself was backed up, it contained only those files that had been backed up at that point in time. It did not contain all the files on the system.

Therefore this phenomenon is a side effect of the normal backup and restore process.

### **After a Full Restore Special Device Files have Different Major and Minor Numbers**

This can occur if you already have a device file present with a major and minor number different from that on the backup tape. When you restore you will see the message...

```
Will not extract because...File exists
```

This is a safety mechanism. If the device file already exists, the major and minor numbers cannot be changed. The file would have to be removed first. However, the permissions and ownerships will be changed to reflect those on the tape.

The special option `-zREPLACE_ALL` option can be used to override this. If specified during a restore, **EDGE** will try to remove files if required before restoring them.

### **An Option in the Manual does not Appear on the Help Screen**

The help screen is designed to show the most commonly used options and what they do. It is not a comprehensive list. There are so many options for the product that this would take several screens. For less commonly used options, you should refer to the written manual.

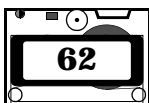
## **5.7 Additional Support Resources**

---

Microlite Corporation maintains an active presence on the **World Wide Web**. Our site at <http://www.microlite.com> contains a support section listing many common problems and their solutions.

Many times, if a problem is encountered which is not described in this manual, a solution can be found on the web site.

Our ftp site (<ftp://ftp.microlite.com>) may also contain fixes, patches, and new releases, and should be checked periodically.



# 6 Error Reference Guide

---

## 6.1 Numeric exit codes

---

Numeric error codes are now maintained in the *BackupEDGE* User's Guide. Please refer to that document.

## 6.2 Fatal Error Messages from `/bin/edge`

---

### FATAL ERRORS WHILE BACKING UP

- `edge: Trying to seek on non-seeking device!!`  
Use `\C` or `\f` option and avoid `\n` option  
Alternatively use `\P` option to force compression on non-seeking device

You have tried to compress files on a device that won't seek. There were errors when trying to seek on the device. You either will need to choose another device that will seek or force no compression with the `C` option. Additionally, you can use the `P` option (Pack regardless) which will compress the files regardless of whether the backup device seeks or not. However, when you use this option you must also include the proper `k` factor (size of the backup device in Kilobytes).

- `edge: Tape Write ERROR 5 occurred because I/O error`  
`Offset is %ld K`  
`Room left is %ld K`

This is probably the most common fatal error that causes **EDGE** to stop. This means the backup device has developed a bad sector or section of tape. This is a hardware media error and means it is time to get new floppy disks or tape cassettes. If this does not resolve the problem, try cleaning the unit or having the hardware sent for maintenance or repair.

- `edge: Tape Write ERROR %d occurred because {reason}`  
`Offset is %ld K`  
`[Bytes actually written: %d ]`  
`Room left is %ld K`

This means a fatal hardware error occurred on the device in question. **EDGE** could not write all the data to the backup device. When using a floppy, this is usually because of a bad sector on the floppy disk. When using a tape, it means a bad section of tape.

If the number of bytes is displayed, only part of the data was successfully written to the backup device. It is possible that the backup device ran out of room before all the data could be written. This could be because the blocking factor is not an even

divisor of the tape size. Simply specify a smaller **k** factor next time and you will have no further problems.

It is also possible that the data is being piped across a network and the network is unreliable. Only part of the data was transferred.

The offset specifies how many kilobytes into the backup this error occurred. If this number is the same no matter what list of files you back up, you have found a bad spot on the backup device at this offset.

The room left signifies the amount of room left on the backup device. If greater than zero, there is still plenty of room to write the data. The default value for room left is a very large number.

- `edge: Hard Read Error %d in file {filename} because {reason}`  
`Do you wish to continue backup? (y/n)`

While backing up the specified file, a **HARDWARE** error occurred. This usually means there is a bad block on the hard disk involving this particular file. The reason is given. Usually this is a serious error and could be a sign of a failing hard disk. If you continue the backup, the following messages will appear...

- `WARNING==>File: {filename} incomplete!!`  
`BAD section filled with 'xxxxx...'`  
`Mark as corrupt!!`

The bad sector of data is replaced with lower-case x's. If this file is later restored, a section of it will contain lower-case x's.

If the backup is being done in the background, a **yes** answer will automatically be supplied. The fact that a file was incompletely backed up will be reflected in the backup summary.

- `{filename}, %d blocks ...compressing====>..failed`  
`ERROR==>File is incomplete!!`  
`Do you wish to continue backup? (y/n)`

This message will be produced if an unreadable block is encountered while compressing a file. If you choose to continue the backup, the compressed file will be truncated at the point where the hard disk read error occurred. When the file is restored it will be shorter than the original version. If the backup is being done in the background, a **yes** answer will automatically be supplied. The fact that a file was incompletely backed up will be reflected in the backup summary.

- `edge: PIPE QUE ERROR - FATAL`  
`Set the environment variable "TMPDIR" to your largest filesystem`

This message will only be produced if the **PIPE\_COMPRESS** environment variable is set and compression is enabled. Normally, software compression uses no temporary storage. However, setting those two options causesThis message means that the Virtual Pipe ran out of space. It usually occurs when compressing a very large file, or just after compressing a large file.

The Virtual Pipe is a pipe that consists of machine memory (the amount depends on your binary) and free space on one of the filesystems. The default filesystem used is the filesystem that contains the directory `/tmp`. Sometimes, this is a small root filesystem that has limited free space. The limited amount of free space was used up by the virtual pipe and it simply ran out of room. **EDGE** requires that there be at least as much free space in the filesystem used as the size of the compressed version of any given file.

The solution to the problem is to instruct **EDGE** to use another filesystem for its Virtual Pipe. The way of doing this is to set the variable `TMPDIR` to a directory that exists on a larger filesystem. For example, if `/dev/u` is a large filesystem with 100 Mb of free space and is mounted on the directory `/u` the command to get **EDGE** to recognize this is...

```
TMPDIR=/u/tmp
export TMPDIR
edge MZvbkf 64 149760 /dev/tape .
```

In this example, it is assumed that the directory `tmp` resides on the filesystem `/dev/u` mounted on `/u`. If it does not exist, then make it first by typing...

```
mkdir /u/tmp
```

If you get the `edge: PIPE QUE ERROR - FATAL` while running an unattended **Differential Backup** or **Master Backup**, or when backing up through **EDGEMENU**, then you can edit the file...

```
/etc/default/edge.cfg
```

This file will have lines of code that set the `TMPDIR` variable for you. Simply change the name of the temporary directory to one which resides in a filesystem with the largest amount of free space.

To verify that you have indeed set the correct filesystem, and that **EDGE** has recognized this, you can run the command `df` while **EDGE** is running and compressing a large file. You should see the available number of blocks decrease on the filesystem you have selected with the `TMPDIR` variable. If you see the root filesystem free space decreasing, you know you did not set the variable correctly or **EDGE** did not recognize it.

Alternatively, unset the `PIPE_COMPRESS` variable to cause **EDGE** to use streaming compression. There is no reason to use `PIPE_COMPRESS` under normal circumstances.

- `traverse: no memory because {reason}`

No more memory is available. This occurred while **EDGE** was traversing the file system. Usually, this occurs when the memory to hold the ever increasing linked file map exceeds the system memory available. The linked file map holds all the files that are linked on the system. The more links that exist, the more memory this requires. To remedy this problem, you can either use a smaller block factor (`b` option) or turn compression off with the `C` option. If neither of these work you may have to increase the maximum memory per process (see XENIX/Unix manual) or add more memory to the hardware.

## FATAL ERRORS WHILE RESTORING OR VERIFYING

- `edge: directory not in proper format`  
`File header (name, size, date) is unobtainable`  
`This means the header information about the file`  
`has been corrupted or is not in the TAR format`

The header section of the file has been corrupted or the previous header file size was wrong. When reading an archive, **EDGE** will first read the header block to find out information about the file including the name, size, modification time, number of links, and whether the file is compressed or not. This information along with a checksum, has been stamped in the header.

The above error message means that the recomputed checksum does not match the checksum stamped into the header when it was created. More often than not, the data block in question is probably not even a header block. This occurs if the wrong volume is inserted in the drive while restoring. Since **EDGE** usually checks volume numbers when restoring this is rare. However, it is possible to get two volume sets mixed up. Also, rarely, someone may have overwritten one of the volumes with other information. You will be allowed a chance for error recovery. **EDGE** will read through the data until it finds a valid header block and then resume restoring from that point on. If **EDGE** is running in the background, it will not prompt for error recovery but will terminate immediately.

`Do you wish to attempt ERROR RECOVERY? (y/n):`

The user is prompted for this message when a hardware error or software error happens during restoring or verifying a backup. If you answer `y`, **EDGE** will attempt to find the next valid file. You will get the message...

`File {filename} may have CORRUPT DATA`

After this, **EDGE** will continue to restore data.

- `edge: {filename}: Serious ERROR restoring data because {reason}`  
This occurs when restoring data to the hard disk from a floppy or tape. **EDGE** has encountered a hardware problem on this hard disk device. This will also occur if the hard disk is out of space. Check the space on the hard disk by using the `df` command. If you have plenty of space, your hard disk is having problems and you will need to map out the bad block in question.
- `edge: Hard Error Reading {device-name} at offset %ld`  
`Error %d occurred because {reason}`  
This is displayed when **EDGE** has difficulty reading the archive while listing files on the archive or while restoring files. This means a serious hardware error exists with the device or with the archive volume.

## FATAL ERRORS GETTING STARTED

- Using file {filename} for file names to archive  
Cannot open...

The file specified with the **F** option could not be opened for reading. Most likely the name was misspelled on the command line.

- A copy of the screen output will go to the catalog file: {filename}  
Cannot open...

The catalog file, where a carbon copy of the screen output will go, cannot be opened for writing. Either a directory in the pathname doesn't exist, a directory in the pathname doesn't allow search permission, or the file already exists and is not writable because of permissions.

- edge: Invalid 'k' volume size==> %s

The volume size specified with the **k** or **s** option is a negative number or cannot be converted into a number (i.e. text). Usually this is because the options and arguments on the command line have been mixed up.

- edge: Invalid 'L' limit==> %s

The limit specified with the **L** option is invalid. The same reasons apply as above.

- edge: Invalid 'b' blocksize==> %d  
edge: Invalid 'g' group number==> %d  
edge: Invalid 'd' Directory depth limit==> %s

These must be positive numbers. The blocksize has a defined upper limit depending on the type of CPU. This limit is shown on the general help display of **EDGE** (displayed when **EDGE** is typed with no command line options).

- edge: tapefile has not been specified

The archive device has not been specified. The **f** option was given but no name of the backup device to use was given.

- You have specified the date incorrectly  
-zDATE=10/4/92-10:30  
-zDATE=10/4/92  
-zDATE=MMddhhmm[yy]  
-zDATE=1004103092  
-zDATE=10041030

The date has been specified incorrectly. The correct usage is shown using examples. There are several ways the date can be specified but the method you chose does not fit any of these. Most likely you reversed month and day-of-the-month. Please refer to "Specifying Dates And Times" on page 38.

- edge: option %s is not supported in this version

The option letter given (%s) may be supported in other versions of **EDGE** or **tar** but not in this one.

- `edge: %s: unknown option`

An unknown option letter has been given to **EDGE**. Please check the command line options and try again.

- `What files did you want to backup?`  
`(No file names were given!)`

This means that the command line option letters were okay, but you did not give **EDGE** any actual files to backup. Unlike some versions of `tar`, **EDGE** will not erase your backup when this happens. A common mistake is to type `edge cv` instead of `edge xv` when restoring. Another common way of doing this is to forget an argument in the command line...

```
edge Cvf insurance.dat
```

In this case, the file `insurance.dat` was meant to be backed up, but the user failed to specify to which device the backup should be sent.

- `edge: 'r' option must use a seeking device (e.g. floppy )`  
If `{device-name}` is really seeking, add the `'n'` option and try again

In order to add files to the end of an archive the device must be seeking. This is because you never know when an archive ends until one has read the last block of data. This block, however, must be rewritten with the new data added to it. Because most tapes and non-seeking devices don't allow a program to skip one block backwards a seeking device must be used.

- `edge: Not enough memory for compress/decompress buffers and blocksize of %d`  
Use a smaller blocksize or turn off compress/decompress with `'C'` option

The combination of memory allocated for the blocksize used and the compress/decompress buffers requires too much memory space. The solutions suggested are self explanatory.

- `edge: Attempt to access {device-name} failed because {reason}`  
Consider:

- 1) Improper USAGE of the `'f'` Option
- 2) Device `'%s'` NOT TURNED ON
- 3) Inadequate PERMISSIONS
- 4) MISSPELLED Device Name `'%s'`
- 5) Device `'%s'` has WRITE PROTECT TAB set
- 6) Write PERMISSION NOT SET for `'%s'`

This means there was trouble accessing the backup device(`%s`) at the start of a backup. The possibilities are self explanatory.

- `edge: Pipe is empty!`

This means that **EDGE** is trying to read data from a pipe but there is no data at all in the pipe. In fact the pipe is empty. Check the program that creates the pipe; it probably aborted prematurely. Most commonly this is due to misspelling of the command or an option.

### **WARNINGS WHILE BACKING UP**

- `edge: can't find {filename} because {reason}`

You have given **EDGE** a list of files to backup but this one specified cannot be found on the hard disk. Usually this is because it is misspelled or it has been removed by another user.
- `Can't open directory {dirname} because {reason}`

While traversing the file system, **EDGE** has come upon a directory that cannot be accessed. Either the directory was removed or the permissions of the directory do not allow the user of **EDGE** to search it.
- `{filename}: WARNING: Maximum path length exceeded!!`

A pathname/filename combination exceeds the maximum number of characters allowed. The maximum number is 400. This may occur on networked systems with long pathnames. It also may occur in a very deeply nested subdirectory. The remedy is to shorten some of the subdirectory names or eliminate the deep nesting. This file is skipped and the backup is continued.
- `Cannot open file ./etc/Master_backup because {reason}`

While doing a **Master Backup**, the above file could not be created because of the reason given. Check permissions.
- `Cannot get ./etc/Master_backup because {reason}`

While attempting to do Differential backup, the date of the last **Master Backup** is obtained from the file `./etc/Master_backup` or `./Master_backup`. Neither of these files exists, which means a **Master Backup** has never been done, the files were removed, or **EDGE** was invoked from a directory other than that from which the **Master Backup** was invoked. Check the directory you are in when **EDGE** was invoked. Generally **Master** and **Differential Backups** should be done from the root directory.
- `^^^^--> LOCKED by another program!! Waiting...`

This means **EDGE** attempted to put an enforced lock on a file that was locked by another program. **EDGE** is waiting for the lock on this file to be released. This message will be displayed for either Unix V or XENIX style locking.
- `Resuming backup...`

This is printed when the lock above has been released and the backup is resuming. It is helpful to know if and when the lock on a large file has been released.

- `^^^^--> LOCKED by another program!! Will retry later...`

This means **EDGE** attempted to place an unenforced lock on a file and couldn't because it was already locked by another program. **EDGE** will try again when finished with the rest of the backup, and if it cannot obtain the lock, it will backup the file anyway.

- `WARNING: File was in use during backup`

Unix versions of **EDGE** print this message when an unenforced lock cannot be placed on a file because it is already locked by another program. It is possible that another program or user could change this file while it is being backed up. Thus, a warning message is printed.

- `WARNING: File was locked (System ENFORCED) will retry later...`

Unix versions will print this message when an unenforced lock cannot be placed on a file because it was already locked by another program or user. However, because the type of lock placed on the file by the other program or user is a System ENFORCED write lock, **EDGE** would hang indefinitely (or until the lock was released) if it tried to backup the file. Rather than do this, a decision was made to try again when all other files have been backed up. Hopefully, at this later time, the System ENFORCED write lock will have been released.

- `XENIX Locking/Unlocking failed because {reason}`  
`UNIX V Locking/Unlocking failed because {reason}`

**EDGE** tries to lock files before backing them up so they won't change. If locking fails you will receive one of the two messages depending on the type of locking available on the system. This is not a critical error, but it does mean that no file locking was done on that particular file while being backed up. If you get many of these messages it would be better to backup in single-user mode.

- `Retrying previous locked files...`

This will appear near the end of a backup. The backup is complete except for those files that could not be backed up because there was lock placed on them. This message means that another attempt will be made to back them up.

- `edge: (2nd lock attempt failed)`  
`File was NOT backed up because lock was ENFORCED!`

This type of message will occur at the end of a backup when previously locked files are attempted to be backed up. A very special type of lock has been placed on the file by another program or user. This type of lock is an ENFORCED write lock. This means that no program can read the file at all. If any program tries to read the file it will hang indefinitely (or until the lock is released). **EDGE** detects this situation and avoids it and informs you of why it did not try to backup the file.

- `edge: (2nd lock attempt failed)`  
`File was backed up WITHOUT a lock!`

This type of message will occur at the end of a backup when previously locked files are attempted to be backed up. An attempt was made again to obtain a lock on the

file, but another program or user already had a lock on the file. Because **Level 1** locking was specified, the file was backed up anyway and this warning message lets you know that no lock had been obtained when the file was backed up.

- **edge: Error opening temporary save-work-file**

A file was locked and could not be accessed. It was determined that this file would be accessed again at the end of the backup. However, when trying to open temporary save-work-file to put the filename into, the open failed. The temporary save-work-file to be opened would reside in the /tmp directory. Make sure that the /tmp directory exists and that you have access permissions for it. On some systems, this directory may not be /tmp depending on the default location for temporary files in the C library routines

This message means that the locked file was not backed up and will not be backed up at the end of the backup.

- **{filename} is not a regular file or directory...**

**EDGE** cannot figure out what kind of file this is. This means the file is not a regular file, directory, block device file, character device file, or named pipe. Several possibilities exist. The file could be a shared memory file, a semaphore file, a named space entry, or a symbolically linked file. If the version of **EDGE** you have was not compiled to handle symbolically linked files, this message will appear when a symbolically linked file is encountered. As Unix/XENIX develops, new types of files may also develop. It is possible that special network implementations in the future will use special files. If your version of **EDGE** is significantly out of date with the version of Unix/XENIX, it is possible that a new file type was added to the version of Unix/XENIX. This message may occur with an upgrade to a new operating system release because of the above reasoning. Also to be considered is the possibility of having a corrupt inode entry on the hard disk. This could make the type of the file unrecognizable.

- **Will not compress...(already ends in "%s" )**

If the filename already ends in a .Z, .z, .zoo, .zip or .arc, **EDGE** assumes the file is in compressed format. Also any other suffixes that were specifically excluded using the **COMP\_EXCL** environment variable will be listed. If **EDGE** tried to compress a previously compressed file the result would most probably be negative compression and a larger file. This feature can be used to advantage when one wants to make sure a certain file is not compressed.

- **edge: file {filename} has been shortened by %ld bytes!!  
Appropriate adjustments made!!**

While **EDGE** was backing up the file, someone shortened it. **EDGE** will adjust for this automatically. If the device is seeking, the shorter length will be used. If the device is non-seeking, **EDGE** will fill in the dead space with nulls (zeros).

- Cannot open file {filename}

This file cannot be opened. The most common cause is the file does not have read permissions. This will happen when **EDGE** is used by a non-super-user. More rarely, a file has been removed by another user between the time **EDGE** saw the file and started to open it for reading.

- edge: cannot open virtual file for backup because {reason}

One of the virtual files specified in the list of virtual files could not be opened. Most commonly a space character was left on the trailing end of the file name in the list. Check the list, delete the name and then retype the name, being careful not to include any spaces. The list of virtual files is specified using the **VIRTUAL\_LIST** environment variable.

- Virtual file backup failed...Invalid header

The accessory program that is used to prepare the virtual file for backup failed to produce a valid header. Usually this means the accessory program is not available or the product was incorrectly installed. Try re-installing the **EDGE** package.

- failed!!

edge: Virtual file backup INCOMPLETE!!

This will show if the virtual file was incompletely backed up. This usually results from the failure of the accessory program that removes the null data from the virtual file. This program is called `/usr/lib/edge/bin/vback`. It failed. Most often this is because the accessory program terminated prematurely or was killed by the super-user.

- edge: UNEXPECTED... NO MORE SPACE on %s

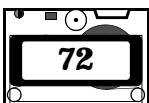
Expected %ld Kilobytes, but ran out of room unexpectedly.  
Please change 'K' option to %ld next time!!  
Recoverable...

This happens when the wrong **k** factor size is given in the command line. For example, if you thought your tape took 40MB you might use a **k** factor of 40960. However, if your tape happened to be the one out of the group of tapes that was 5% shorter you would get this error. Experience has shown that the exact capacity of the tape may vary up to 10% of what is actually claimed. Additionally, even tapes with the same capacity will vary from one to another.

Two results are possible. If the word **Recoverable** appears, the backup is still good and you do not need to repeat it. **EDGE** has ways of recognizing this situation and can recover. If this message is not present then the backup is bad. Either way, **EDGE** will suggest the proper value to be used next time so that this won't happen again.

- getcwd: trouble opening pipe...failed

This warning message occurs when **EDGE** is checking to see which directory you are in when **EDGE** is invoked. This will only occur on Berkeley systems where `getcwd.o` (get current working directory) is not part of the standard C library.



- `findlink: not enough memory for linkmap`

As **EDGE** backs up files it creates an internal map of what files are linked to other files. This map grows larger as the backup continues. The size of the map has exceeded memory limitations. The solution is to increase the memory allotted to a single process, manually decrease the number of linked files, or split the backup into two smaller backups of the same data.

- `Findlink: Error reading link file occurred because {reason}`  
`Findlink: Error writing link file occurred because {reason}`

These errors will only appear in the small model compiled version of **EDGE**. An error occurred reading a temporary file that holds the map of linked files. The reason is given. You can tell if your version of **EDGE** is small model compiled by using the `file` command on the Unix/XENIX system. You can also use the `hdr` command.

- `a {filename} (Special -zDEV ), Not backed up!!`  
`edge: {filename} must be a character device`

This will be generated when the `-zDEV` option is used to back up special raw device files. One of the devices specified in fact is not a character device. The raw device backup feature must use a character device.

- `Special '-z' option: "DEV" - RAW Device Backup`  
This confirms the correct use of the `-zDEV` option.

- `a {filename} (Special -zDEV ), Not backed up!!`  
`edge: {filename} must be a character device`

The `-zDEV` option was used to backup a raw device. The filename given, however, was not actually a character device so a raw partition backup could not be done. This is not fatal, but the file is not backed up.

---

## WARNINGS WHILE RESTORING OR VERIFYING DATA

- **ERROR DURING VERIFICATION!!**

This will occur at the end of a verification when the `T` option is used if errors were encountered. In most but not all cases the reason for the error can be determined by scanning the Catalog file for the `edge:` identifier. Many times the verification required **ERROR RECOVERY** mode to get through some bad spots in the media. This will also show up for any internal error at all. It is very precise so it is possible that minor errors that do not represent any change in the integrity of the data could result in this error.

- `Wrong volume==> %d (Looking for %d) Try again...`

This is self explanatory. You will only get this message two times. After that, **EDGE** assumes you know what you are doing and really want to restore the volumes out of

order. It is possible that one volume of a set was damaged (coffee spill), and you need to restore out of order. So, this message can be overridden if you are persistent.

- `decompress: cannot open output stream because {reason}`

An error occurred while initiating the streaming decompress for this particular file. An attempt to create a stream with write permission failed. Check the permissions of the file being restored or re-invoke the restore with super-user power.

- `{filename}: not in compressed format`

This message is given by the decompression part of **EDGE** when asked to decompress a file that is really not in compressed format. **EDGE** determines if a file is in compressed format by two tests.

a. The header block of the file signifies the file is compressed

b. The first two bytes of the file have been stamped with a special value signifying the file is compressed.

This message will occur if you start restoring on a volume other than the first and you have a split compressed file across volumes. Even though the name and header will indicate that the file is compressed, the first two bytes will not have the special value signifying compressed format. These two special bytes will be on the first part of the split file.

Try putting a previous volume in to see if the file actually does start on the earlier volume.

- `{filename}: compressed with %d bits, can only handle %d bits`

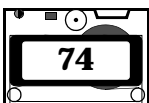
Here %d represents an integer between 9 and 16. This file was compressed with a bit value larger than this version can handle. The larger the bit value used, the more memory consumed by the program. This can happen if you take a floppy disk that has compressed files in 13 bit format (most versions) and try to restore these to a machine where **EDGE** was small model compiled (ALTOS). The small model version can only decompress files using 12 bits so you are out of luck.

- `decompress: Error writing to file because {reason}`

An error occurred writing the decompressed data into the file on the hard disk. Most commonly this is because you have run out of space on the hard disk filesystem. Free some more space on the hard disk filesystem and try again. Occasionally, the error is due to a hard disk fault. This is a serious error and means the file was not restored properly.

- `Decompress: Out of memory`

This message will only occur on versions of **EDGE** compiled by using a small memory model option. This is done for computers with an extremely small amount of memory. This is because the memory is so scarce that it is allocated when needed.



You can try using a smaller block size (try 2) to free some more memory. This message is primarily of historical significance since the small memory model is not used anymore.

- `Expanding===> encountered corrupt data...failed!!`

This means that as the file was decompressed, corrupt data was encountered and it could not be decompressed further. The file is restored up to the point where the corrupt data was encountered and then truncated. Usually there will be some minor corruption near the tail end of the file.

The cause of this is usually a hardware problem during the original backup or during the restore. Also, the media could have been corrupted by electromagnetic radiation after it was backed up. It is also possible on seeking devices, that parts of the media were overwritten by another program. Additionally, it could be that the media was backed up using an error-correcting device such as `/dev/erct0` and yet restored using a non-error-correcting device such as `/dev/rct0`.

This is a tip-of-the-iceberg type of problem. There is probably widespread corruption on the tape media. Be aware of the fact that there is probably some corruption in non-compressed files that were restored during the same session. You should check out all restored files carefully, even if there were no error messages.

You should check to be sure you are using the same blocking factor and capacity and device and try again. If problems are still encountered in the same spot, chances are you have a hardware problem with the tape drive or a media problem with the tape.

- `{filename} (Virtual File)`  
`Revirtualizing===> ...failed!!`

This will only occur on versions that support Virtual files. The accessory program that revirtualizes the file has failed. The file is incompletely restored. Please check that you have `/usr/lib/edge/bin/vrestore` on the system and that it has not been corrupted.

- `WARNING: tovfile() - buffer at limit`

This will only happen when revirtualizing virtual files. It means an internal error occurred. An internal buffer problem occurred and appropriate corrections were made to correct this problem. However, you should call technical support and review the circumstances leading to this problem.

- **Extraction Warning Messages:**  
`edge: {filename}: cannot link because {reason}`  
`edge: Symbolic link failed because {reason}`  
`edge: Cannot create {filename} because {reason}`  
`Will not extract because...{reason}`

These are informative messages printed out while restoring files to clarify why certain actions were not taken. They are all self explanatory.

- `edge: cannot make directory path because {reason}`

**EDGE** has tried to make the necessary path to restore a given `pathname/{filename}`. The attempt to make one directory in the path failed for the given reason. Thus, the file was not restored successfully.

- `Directory not made because...{reason}`

This is printed when doing a restore to inform the user that the directory was not created because of the reason given.

- `Will link to --> {linkname} ... failed`  
`edge: You must restore {filename} first!!`

A selective restore of a few files was performed and one of these was a linked file. The file that it is linked to, however, does not exist on the hard disk. Therefore, this file needs to be restored first. The restore should be retried and the file `{filename}` given should be included in the new list of files to be restored.

- `Will link to --> {filename} ... failed!`  
`edge: WARNING: A Symbolic link was used instead`

This will happen if you are going from one filesystem to two filesystems and a regular link cannot be used because it will span across the two different filesystems. In this case, a symbolic link is conveniently used instead. In the restore summary, this file will still show as incompletely restored.

- `Will link to --> {filename} ... failed!`  
`edge: {filename}: cannot link because {reason}`

**EDGE** attempted to restore the link and it failed because of the above reason. This most commonly occurs when you are taking files from a system with one large filesystem to a system with two or more filesystems. A link which linked two files on the same filesystem is attempted to be restored to a system that requires that the two files be in different filesystems. It is impossible for a regular link to span two filesystems so this will fail.

- `edge: Symbolic link not supported on this computer`  
`edge: WARNING: A regular link was used instead`

This occurs when you are restoring an archive from another computer that supports symbolic links and trying to restore it to a computer that does not support symbolic links. In this case, a regular link was used instead of a symbolic link. and the restore summary will show this file as incompletely restored. This should alert you to look for the above WARNING message in the catalog file.

- `edge: Symbolic link not supported on this computer`

Similar to the above message except that a regular link could not be used. Most likely this is because the previous symbolic link was to files across different filesystems. **EDGE** attempted to use a regular link instead but discovered that it could not be done because a regular link cannot span filesystems.

- `edge: Symbolic link not supported this version`  
`WARNING: A regular link was used instead`

The binary version of **EDGE** running does not internally support symbolic links. This is usually because the operating system under which it is running also does not support symbolic links. Therefore a regular link was made instead. This will show up in the restore summary as a file that was incompletely restored.

- `edge: Symbolic link not supported this version`

The binary version of **EDGE** running does not internally support symbolic links. This is usually because the operating system under which **EDGE** is running also does not support symbolic links. A regular link could not be made because the link would span across two different filesystems.

- `WARNING: About to restore extent #&d out of order!`  
`Proceed (y/n)?`

If, for whatever reason, a file that is split across multiple volumes is restored out of order, you will receive this message. Sometimes it is desirable to do this. For example, if a file is split across several floppy disks, and one of the diskettes is severely damaged (or missing), you will want to restore what you can. If you answer `n`, the remainder of this file will be skipped.

- `DOS textmode conversion...`

This message will occur when restoring files backed up on a DOS system with a DOS compatible `tar` utility and restoring to a Unix system. The DOS files (if they are text only files) will be converted to the Unix format. All carriage-returns preceded by a linefeed are converted to linefeed alone. If you do not want any conversions done, you can use the `-zNOCNVT` option.

If the file is binary, you should not see this message. The decision to convert is made by inspecting the first 512 characters of the file. If there are non-ascii or non-printable characters the file is considered binary.

- `*** End of Edge Archive ***`  
`Do you wish to continue ERROR RECOVERY? (y/n)`

This will occur when you are restoring or listing and have already encountered and entered `ERROR RECOVERY` mode. **EDGE** believes that you have reached the true end of the archive. However, if you are doing difficult error recovery work and need to list or obtain some valid `tar` files beyond this point (if they exist), you are given this option. In most cases, you would answer `n` to this question.

---

## WARNINGS GETTING STARTED

- `Special '-z' option: "%s" - Not recognized or implemented!`

A `-z` option was entered that is not in the list supported. Most likely the name of the option was misspelled. The spelling has to be exact, and getting upper-case and lower- case letters correct is important.

- `Excluding ==> %s`  
This will be repeated for files or directories excluded on a restore. There is no distinction made between files or directories in this listing. If a backup is being done, this will represent the directories that are excluded.
- `Excluding file ==> {filename}`  
This will be repeated for each file excluded during a backup.
- `Excluding wild-card pattern ==> %s`  
This is repeated for each wild-card pattern excluded either during a backup or restore.
- `Speed option for Double buffering in effect`  
This confirms that Double buffering will be used. This message also appears in the catalog file.
- `Running as a background task`  
This indicates that **EDGE** believes it is running as a background task. The behavior of **EDGE** is different and these differences can be reviewed in the **EDGE Manual** pages under “Background Operation” on page 34.
- `Increasing ulimit from %ld to %ld for large capacity drive`  
The system ulimit needed to be increased to match the large capacity tape drive you are using.
- `Increasing ulimit from %ld to %ld for large capacity drive failed!!  
The reason was {reason}`  
An attempt was made to increase the ulimit but it failed. This will occur if the user is not the super-user. It is a warning that the backup may stop when the system ulimit is reached.
- `ulimit() system call failed because {reason}`  
An attempt was made to obtain the system ulimit value. This attempt failed for the reason given.
- `Using Unenforced File Locking`  
This confirms that unenforced file locking will be used during the backup.
- `Using ENFORCED File Locking`  
This confirms that enforced locking will be used during the backup. This means that each file backed up is guaranteed to be locked against writing before it is backed up.
- `Cannot get all default parameters for device: %d [0-99]  
Check permissions or structure of /etc/default/edge`  
An attempt was made to read the blocksize, device name, and capacity from either `/etc/default/edge` or `/etc/default/tar` and this could not be done. Either there is no permission to access the file or it is corrupt.

- Please edit `/etc/default/tar` and specify size (in K) of device: `%d [0-9]`

The capacity for this device number was set to zero. It needs to be set to a physical capacity in kilobytes. Although some versions of `tar` allow a capacity of 0, **EDGE** does not because of compression constraints.

## 6.4 Backup Summary Messages

---

- `****> ERROR: %d File(s) NOT backed up!!`

This means the specified number of files were not backed up. This could be for many reasons. The most common reason for a large number of files not being backed up is that the backup was done by an account other than super-user. Thus, only the files that person has privileges for can be backed up. Other reasons include inability to open a file (perhaps across an NFS network), a file not being found by that spelling of the name, the name of a file is greater than the maximum allowable **EDGE** path length, inability to lock a file because enforced locking was used by another process, information about the file cannot be obtained (size, date, etc.), or the file type cannot be determined. The Catalog file should be scanned for error messages starting with `edge:` to determine the exact reason files could not be backed up.

- `****> ERROR: %d File(s) INCOMPLETELY backed up!!`  
`****> BAD Data Blocks on the Hard Disk were encountered!!`

This means that hardware errors were encountered during the backup. Specifically, there were defective data blocks encountered on the hard disk drive and the files involved were only partially backed up. Again, in this case, you should scan the Catalog file for details of what happened and the names of the files involved.

## 6.5 Restore Summary Messages

---

- `FILES successfully restored: %d`

This represents the number of files that were successfully restored. This means that no errors were encountered during the restore.

- `WARNING: %d file(s) NOT restored!`  
`Files Requested: %d Files Restored: %d`

This represents the number of files requested to be restored less the actual number that were restored. Check the spelling of the files that weren't restored. You can see if a file by that spelling exists in the Catalog file representing the original backup for that medium. Alternatively, you can list the entire contents of the medium using the `tV` option and then see if the file is listed by looking for it in the Catalog file.

- **WARNING: %d files INCOMPLETELY restored!**

This represents the number of files that were incompletely restored. There are many reasons for this, including:

- a) Incomplete decompression of a file due to corruption
- b) Incomplete revirtualization of a virtual file
- c) A regular link that spanned filesystems thus requiring symbolic link
- d) A symbolic link (from another system) converted to a regular link
- e) A regular file partially restored using ERROR RECOVERY

## 6.6 *Miscellaneous Error Messages*

---

- **edge: Running as background task and cannot get a reply**

An attempt was made to look for a reply in the named pipe `/dev/edge_listen` and this failed. Most commonly, it is because this named pipe is not present on the system. This will usually occur while crossing volumes. You can create the named pipe by using the command...

```
mknod /dev/edge_listen p
```

If the named pipe is already present, check the access permissions since it is possible that the permissions on the named pipe are too restrictive.

- **edge: Can't open your terminal for reading**

An attempt was made to read from your terminal, but it could not be opened for reading. Check the access permissions of `/dev/tty` and make sure that it is a character device. In rare cases, it is possible that the standard input appears to be a terminal to the system but cannot be opened for reading. This message can appear when **EDGE** is crossing volumes and trying to obtain the answer to the volume prompt message.

- **Can't open /dev/tty**

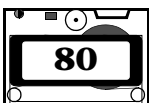
The device file `/dev/tty` could not be opened. Check the access permissions of this character device. If this message appears, you also need write access permission to this device.

## 6.7 *Network Error Messages*

---

```
Remote site: %s with tape device of %s failed to respond because  
{reason}
```

An attempt was made to establish a remote link to the tape device. This attempt failed for the given reason. Usually, this is a network permission problem. The user invoking the program does not have proper permission to make the link across the network.



If the environment variables `REMOTE_PUT` or `REMOTE_GET` have been set, examine them and make sure they can execute independently. When you execute them independently, you may see the reason for the failure.

If you are certain that you do not have a permission problem accessing the network, then it is probable that the default method **EDGE** uses to establish the network link will not work. You will have to set the environment variables `REMOTE_PUT` or `REMOTE_GET` to the appropriate remote command to establish a network link.

See the section in the UNIX Manual entitled “**ENVIRONMENT VARIABLES**” and “**NETWORKING**” for details on how to do this.

```
edge: NETWORK LINK just FAILED!!  
Command was: "%s"  
Must abort...
```

The network link to the remote tape device failed while backing up data. In this case the link was suddenly disconnected and the reason for it is unclear. There could be problems with the network. The remote machine with the tape drive could have gone down or crashed. If this occurs after data has been transferring for a while, it is a very serious problem. If it occurs immediately after starting, it means that a link to the remote device simply cannot be made. The command that was used to establish the network link is given. Check this command carefully for reasons it may have failed. If the command needs to be altered it can be done by changing the environment variable `REMOTE_PUT`.

```
edge: Remote Pipe failed because {reason}
```

While restoring data or listing files from a remote tape, the network link failed for the reason given. If this happened immediately with no files listed or restored, then you can try a different (usually smaller) block factor. Additionally, if you have set the environment variable `REMOTE_GET`, you should examine it and consider changing it.

If this happened after files have been listed or restored, there is a definite problem with the network connection. This may be due to various reasons, such as a “down system” at the remote site.

It is also possible to get this error message if you are using a local pipe from which to obtain archive data and you have the networking version of the product. In this case, you should try a different (smaller) block size.

```
edge: Hard Read Error 13 in file {filename} because  
...Permission denied
```

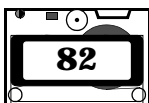
This will occur when trying to back up a file that was NFS mounted and the file does not have read permission for “others”. In addition to the above message, you will be prompted to go into error recovery.

This is a significant problem and is because of the way NFS handles security across networks. When the root user goes to access a file on a NFS network, it will only have the equivalent power of the user called “nobody”. This means that unless the file has read/write permissions for “others”, it will not be able to read the file.

Therefore, **EDGE** will return an **error 13: Permission denied**. The solution to this problem is to ensure that the **root** user has peer permission when doing a backup.

For Unix System V variants, this is done using the **/etc/exports** file and the **exportfs** command. For Sun/BSD variants, this is done using the **/etc/dfs/dfstab** file and the **share/unshare** commands.

Please see your operating system manuals for appropriate examples.



## 7 Index

---

### Symbols

/bin/Edge • 47  
/bin/edge • 11  
/dev/edge\_listen • 77  
/etc/Master\_backup • 57  
/usr/lib/edge/bin/vback • 70  
/usr/lib/edge/bin/vrestore • 72

### B

BackupEDGE  
    New Features • 5  
    Terms • 6

### C

Choose • 9  
COMP\_EXCL • 53, 69

### D

Decryption • 25

### E

Edge • 47  
edge • 11  
EDGE CAT • 52  
EDGE FILE • 52, 57  
Encryption • 25  
Error Reference • 61

### F

FATAL ERRORS  
    Getting Started • 64  
    While Backing Up • 61  
    While Restoring Or Verifying • 63

### H

http  
    //www.microlite.com • 59

### M

Miscellaneous Tips • 51

### N

Notational Conventions • 9

### S

Select • 9  
SUFFIXES • 53

### T

Tapes  
    4mm or 8mm • 53  
    Byte Swapped Data • 54  
Terms Used in Manual • 6–??  
TMPDIR • 62

### V

VIRTUAL\_LIST • 69

### W

WARNINGS  
    Getting Started • 75  
    While Backing Up • 66  
    While Restoring Or Verifying • 71

# ***Index***

---

